UNIVERSITY OF MANNHEIM
School of Business Informatics and Mathematics

Chair of Practical Computer Science IV:
Dependable Systems Engineering

Master's Thesis

Vulnerabilities in Privacy-Preserving Record Linkage: The Threat of Dataset Extension Attacks

as part of the degree program Master of Science Business Informatics submitted by

Marcel Mildenberger
Matriculation number 1979905

on July 26, 2025.

Supervisor: Prof. Dr. Frederik Armknecht
PhD Student Jochen Schäfer

Abstract

The abstract should serve as an independent piece of information on your Thesis conveying a concise description of the main aspects and most important results. It should not be excessively long.

Contents

List of Figures

List of Tables

List of Algorithms

List of Code Snippets

Acronyms

ANN Artificial Neural Network

BF Bloom Filter

DAG Directed Acyclic Graph

DEA Dataset Extension Attack

DFS Depth First Search

ELD Encoded Linkage Data

GMA Graph Matching Attack

LLM Large Language Model

ML Machine Learning

PII Personally Identifiable Information

PPRL Privacy-Preserving Record Linkage

PRNG Pseudo-Random Number Generator

SMC Secure Multi-Party Computation

TMH Tabulation MinHash

TSH Two-Step Hash

1. Introduction

Linking data and records is an important component of research, software development and software projects. The primary reason for integrating data from different sources is to gain richer, more comprehensive insights about the same entity. Initially, deterministic record linkage, which relies on exact matches between predefined identifiers such as unique IDs, was the main method used in early linkage techniques. However, deterministic approaches often fail in real-world scenarios where data may suffer from inconsistent formatting, typographical errors or missing values, making exact matches impossible [HSW07].

The introduction of a probabilistic framework for record linkage by Fellegi and Sunter in 1969 [FS69] marked a significant advancement in overcoming the limitations of earlier deterministic approaches. In their seminal work, "A Theory for Record Linkage", they proposed a statistical model that calculates the probability that two records refer to the same entity, even in the presence of inconsistencies or missing data. The model evaluates agreement and disagreement patterns across selected attributes and assigns weights based on the likelihood of a true match or non-match. By systematically accounting for real world data variability, the Fellegi–Sunter model has become a foundational methodology for data linkage, particularly in heterogeneous or distributed environments where exact matching is often infeasible [FS69].

Such a probabilistic approach to record linkage is important in sectors such as healthcare and social sciences, where data is often distributed across multiple institutions or sources and lacks unique identifiers. In these fields, the ability to integrate datasets is essential for gaining insights and improving outcomes. In the United States, for example, the healthcare system is highly fragmented, consisting of numerous independent entities such as hospitals, clinics, insurance companies, public health agencies, and research institutions. Each of these organisations collects and stores patient data independently, often using different systems and, more importantly, different formats. This fragmentation creates significant challenges when trying to track patient outcomes, monitor disease outbreaks, or evaluate the effectiveness of treatments across populations. Effective data linkage can bridge these gaps by linking records that refer to the same individual across multiple datasets. This integration is critical for tasks such as epidemiological research, public health surveillance, personalised medicine and healthcare quality improvements [PSZ+24; VSCR17].

For example, during the COVID-19 pandemic, the inability to efficiently link data between testing centres, hospitals and vaccination sites limited timely tracking of infection rates and vaccination outcomes. Had more robust data linkage mechanisms been in place, public health officials could have responded more effectively to outbreaks and targeted interventions to specific populations. Data linkage thus plays a critical role in transforming fragmented data landscapes into unified and actionable insights, leading to more informed decision-making. In response to the COVID-19 pandemic, organisations such as the Centers for Disease Control and Prevention and the Food and Drug Administration have launched projects to address these challenges and further develop linkage techniques [PSZ+24].

In scenarios such as the COVID-19 pandemic, data integration efforts often involve linking records of natural persons from multiple sources. For example, integrating data from differ-

ent healthcare providers, laboratories and public health agencies typically requires the use of pseudo-identifiers derived from Personally Identifiable Information (PII), such as names, dates of birth or other sensitive information. However, reliance on PII for linkage raises significant privacy concerns, as improper handling of such data can lead to re-identification of individuals, with potentially serious consequences such as data breaches, identity theft or unauthorised access to personal health information [PSZ+24; SBR09].

The increasing digitisation of personal data has already led to large-scale data breaches, demonstrating the risks of improperly secured data. Notable incidents such as the Cambridge Analytica scandal, in which personal data was misused for political profiling, highlights the ethical and regulatory challenges of data integration [IH18]. Similarly, healthcare data leaks have raised concerns about the implications of unauthorised access to medical histories, genetic data and insurance records. Leaks of personal health information can have serious consequences, including blackmail, discrimination, and fraud, which can cause significant personal harm. For example, individuals whose medical histories are exposed may face discrimination in employment or insurance, while others may become targets of scams that exploit their health conditions. The potential for such abuse underlines the critical importance of robust data protection measures by working with PII [Smi16].

To address these privacy risks, various techniques have been developed to protect PII during the linking process, primarily by encoding the data prior to linking. However, the use of encoded PII as pseudo-identifiers presents additional challenges. The key question is how to efficiently encode sensitive information while maintaining the ability to accurately match records [SBR09].

Therefore, Privacy-Preserving Record Linkage (PPRL) techniques are designed to facilitate data integration without exposing sensitive information, ensuring that datasets can be securely linked across different entities. To enable linkage while preserving privacy, similarity preserving encoding is applied to the PII. Without such similarity preserving encoding, matches between encoded entities in different databases would not be possible [SBR09; VSCR17]. Over time, three main privacy-preserving encoding schemes have emerged as enablers for PPRL [SAH24; VCRS20].

Bloom Filter (BF) encoding is the most widely used technique in PPRL and is often considered the reference standard [SAH24]. Originally introduced by Burton Bloom in 1970 as a probabilistic data structure for efficient set membership testing [Blo70], BFs were later adapted for PPRL due to their simplicity and efficiency in both storing and computing set similarities. Their compact representation and probabilistic nature make them ideal for scalable PPRL systems, especially in environments dealing with large datasets [SBR09]. The seminal work of Schnell et al. [SBR09] demonstrated the use of BFs in PPRL, particularly in healthcare, highlighting their ability to perform secure record matching without exposing sensitive identifiers [SBR09].

However, BFs are not without limitations. Their vulnerability to graph based attacks and pattern exploitation has driven research into improving their security. Techniques such as diffusion have been proposed to obscure recognisable patterns and increase security [AHS23; SAH24]. For example, Armknecht et al. [AHS23] explored methods to strengthen the security of BF by adding a linear diffusion layer to the BF based PPRL approach, which complicates pattern mining attacks.

To address some of these weaknesses of BFs, Tabulation MinHash (TMH) encoding has been introduced as a more secure alternative. MinHash, first developed by Broder in 1997 for estimating set similarities in large document collections [Bro97], has been adapted using

tabulation based hashing [Smi17]. Although less widely used than BFs, TMH offers distinct advantages, including stronger security guarantees against re-identification attacks. However, these benefits come at the cost of increased computational complexity and memory usage, which may limit its applicability in resource-constrained environments [Smi17].

A further development in encoding techniques is the introduction of Two-Step Hash (TSH) encoding, which aims to combine the strengths of both BFs and TMH while mitigating their respective weaknesses. As detailed by [RCS20], TSH employs a two-stage process. Data is first encoded using multiple BFs, followed by an additional hashing layer that transforms the encoded data into a set of integers suitable for similarity comparison. This layered approach enhances privacy by adding an extra layer of obfuscation, making it more resistant to attacks, while maintaining efficient similarity computations [RCS20; VCRS20].

In practice, BF based PPRL has become the dominant standard and is widely used in areas such as crime detection, fraud prevention and national security due to its balance of efficiency and ease of implementation. However, BF based PPRL systems are not without limitations and vulnerabilities. Previous research has shown that there are several attacks targeting PPRL systems, with a focus on exploiting the weaknesses inherent in BF encodings. These attacks specifically target weaknesses in BF constructions, such as the weaknesses introduced by possible double hashing, structural flaws in filter design, and susceptibility to common pattern-mining techniques. Notably, no specific attacks have been developed for TMH or TSH encodings, suggesting that research has focused primarily on the more widely used BF scheme [VCRS20].

However, a more recent and practical attack has emerged that exploits vulnerabilities common to all PPRL encoding schemes. The Graph Matching Attack (GMA) uses publicly available data, such as telephone directories, to re-identify encoded individuals based on overlapping records between plaintext and encoded databases [SAH24; VCRS20]. Unlike previous attacks that focus solely on the encoding scheme of BFs, the GMA works independently of the encoding scheme chosen. It therefore exploits the graph structure of encoded datasets to re-identify records. Given two datasets, a plaintext reference dataset and an encoded dataset, an attacker can construct similarity graphs where nodes represent individuals and edges represent similarity scores. By solving a graph isomorphism problem, attackers can infer one-to-one mappings between encoded and plaintext records, effectively breaking the privacy guarantees of PPRL. The effectiveness of GMAs depends on the overlap between the two sets of data. The greater the overlap, the higher the probability of successful re-identification. While GMAs can successfully re-identify individuals present in both the plaintext and encoded datasets, their effectiveness is limited to the overlapping subset of the two databases [SAH24; VCRS20].

This work aims to go beyond traditional GMAs by re-identifying not only individuals present in the overlapping datasets, but as many individuals as possible from the encoded PPRL data. To achieve this, the newly introduced Dataset Extension Attack (DEA) builds on the foundations laid by GMAs. The DEA uses an Artificial Neural Network (ANN) trained on the subset of previously re-identified individuals to predict and decode the remaining encoded records. In doing so, the DEA significantly expands the scope and effectiveness of the attack, enabling broader de-anonymisation of PPRL datasets beyond the limitations of existing graph based methods.

1.1. Motivation

The increasing use of PPRL in highly sensitive areas such as healthcare, finance and national security requires research to validate existing techniques and ensure robust privacy [SBR09]. As data-driven applications continue to evolve, the complexity and volume of data being collected and linked across multiple sources is growing rapidly. While PPRL systems are designed to facilitate secure data integration without compromising privacy, evolving cybersecurity threats and attack techniques highlight the urgent need to reassess the resilience of these systems [VSCR17].

Privacy has always been a critical concern in data management, but its importance has been intensified in the era of Artifical Intelligence and Machine Learning (ML). These technologies increasingly rely on large data sets, often containing sensitive PII such as medical records, financial transactions or behavioural data for training. If compromised, the exposure of such data can lead to serious privacy violations, including identity theft, financial fraud and discrimination. The rise of data brokerage, where personal information is collected, aggregated and sold, often without explicit user consent, further exacerbates privacy concerns. This commoditisation of personal data has made PII an attractive target for malicious actors, increasing the risk of unauthorised data linking and re-identification attacks. As ML models become more advanced, the demand for rich, high-quality data continues to grow, making privacy an increasingly pressing issue [KM24; MK19].

In this context, the vulnerability of PPRL systems to emerging attack methods is of particular concern. While PPRL techniques such as BF are designed to hide sensitive identifiers during the data linkage process, recent research has shown that these systems are vulnerable to GMAs. GMAs exploit the similarity preserving properties of common encoding schemes to re-identify individuals by comparing patterns in encoded records with those in publicly available plaintext records. This approach undermines the fundamental goal of PPRL, to protect sensitive data during the record linkage process. Although GMAs are limited to re-identifying individuals present in both the encoded and plaintext datasets, even partial data exposure in highly sensitive areas can have serious consequences [SAH24; VCRS20].

The introduction of DEAs poses an even greater threat to the integrity of PPRL systems. Unlike GMAs, DEAs aim to extend the scope of re-identification to as many individuals as possible within the encoded database. Using ANNs trained on previously decoded data from GMAs, DEAs can predict and decode additional records, potentially leading to the furhter deanonymisation of encoded records. This represents a paradigm shift, as it challenges the viability of widely used PPRL techniques, such as BF based encoding, which have been considered secure.

The primary motivation for this research is to proactively explore and empirically demonstrate the risks posed by advanced inference attacks, with the goal of mitigating their realization in real world applications. By revealing the potential vulnerabilities inherent in current PPRL systems, this work illustrates how adversaries could exploit partially decrypted or encoded data to compromise individual privacy at scale. A successful implementation of the DEA provides concrete evidence that existing state-of-the-art methods lack sufficient resilience, thereby underscoring the urgent need for more robust and secure privacy-preserving techniques.

Furthermore, there is a notable gap in current research regarding the extension of attack capabilities beyond the intersection of datasets. While significant efforts have been made to address the vulnerabilities exposed by GMAs, there is a lack of comprehensive studies exploring how ML can be used to generalise these attacks and compromise further records. This research

aims to fill this gap by developing and evaluating the DEA, thereby contributing to a broader understanding of PPRL vulnerabilities.

By addressing this gap, this thesis aims to contribute to the body of knowledge on PPRL vulnerabilities and serve as a foundation for future research aimed at strengthening these systems. The knowledge gained from this study will help to enable the development of more secure PPRL techniques.

1.2. Related Work

The study by Vidanage et al. [VCRS20] represents a significant advancement in the field of PPRL through the introduction of a new attack method known as GMA. Their work begins with a comprehensive overview of PPRL systems and the similarity preserving encoding techniques commonly used, such as BFs. The GMA exploits weaknesses in these encoding schemes by exploiting their ability to preserve partial similarity information even after encoding. By constructing similarity graphs from both encoded and plaintext datasets, the GMA solves a graph isomorphism problem to align nodes and successfully re-identify individuals in the encoded dataset using publicly available sources such as telephone directories. This method demonstrates the universal applicability of GMAs across different PPRL schemes, and highlights a weakness in systems previously thought to be more robust [VCRS20].

Building on this foundation, Schäfer et al. [SAH24] revisited and extended the work of Vidanage et al. Their contribution lies in a reproduction and replication of the original GMA, during which they identified a flaw, an undocumented pre processing step in the provided codebase that inadvertently increased the effectiveness of the attack. While this step was originally intended to improve computational performance, it introduced errors into the proposed GMA. Schäfer et al. [SAH24] corrected this problem and further optimised the GMA, resulting in improved robustness and efficiency. Their improved implementation achieved higher re-identification rates compared to the original approach. These improvements not only validate the vulnerabilities highlighted by the GMA, but also highlight the potential for refining attack methodologies to expose even greater weaknesses in PPRL systems.

The work of Schäfer et al. [SAH24] is particularly relevant to this thesis, as their improved GMA implementation and accompanying codebase form the basis of the DEA proposed in this study. While the GMA is limited to re-identifying individuals present in both encoded and plaintext datasets, the DEA seeks to extend the scope of re-identification beyond this intersection. Using ANNs trained on the re-identified individuals from the GMA, the DEA aims to predict and decode additional records, potentially leading to complete de-anonymisation of encoded datasets.

To date, no existing research has proposed an approach comparable to the DEA. This thesis addresses this gap by developing and evaluating the DEA, thereby contributing to a broader understanding of PPRL vulnerabilities and highlighting the urgent need for more secure data linking techniques.

1.3. Contribution

The contribution of this thesis is divided into three main parts. First, a comprehensive analysis of PPRL systems is carried out, with particular emphasis on the three main encoding schemes: BF encoding, TSH encoding and TMH encoding. This analysis aims to highlight the

basic principles, strengths and weaknesses of each encoding scheme, setting the stage for the subsequent investigation of their susceptibility to the DEA.

Next, the current state of the art GMA is analysed and its limitations are discussed in detail. Although GMAs have proven effective in re-identifying individuals within overlapping datasets, their applicability is limited to the intersection of plaintext and encoded records. This limitation highlights the need for more advanced attack strategies that can go beyond this.

The main focus of this thesis is the implementation and evaluation of the DEA, which attempts to outperform the capabilities of GMAs by decoding a larger fraction of encoded records. To achieve this, this thesis examines the conceptual foundations, theoretical underpinnings and technical requirements of the DEA. Building on the initial re-identifications made by the GMA, the DEA employs a supervised machine learning approach, specifically using ANNs trained on previously decoded data to predict and re-identify remaining encoded records. This method significantly extends the scope of de-anonymisation in PPRL systems and provides a novel approach to current research.

The DEA is then evaluated against the three main PPRL encoding schemes. While the specific encoding scheme has minimal impact on the GMA, which is primarily based on solving a graph isomorphism problem, it plays a role in the DEA. This is due to the fact that the ANN has to be trained separately for each encoding scheme to account for the unique structural features and nuances of the encoding. However, the DEA is designed with adaptability in mind, ensuring that it can be applied across different encoding schemes, thus increasing its generalisability and practical relevance.

Through this research, this thesis aims to answer critical questions about the robustness of PPRL systems. It investigates how effective supervised machine learning based DEAs are at re-identifying the remaining entries that GMAs cannot decode. It also examines how different encoding schemes affect the performance and accuracy of the DEA, providing insight into which schemes are more susceptible to such attacks and why. By addressing these issues, this thesis contributes to a deeper understanding of the vulnerabilities inherent in PPRL systems and lays the groundwork for the development of more secure privacy preserving techniques.

1.4. Organization of this Thesis

This thesis is divided into four main sections: technical background, methodology, results, and conclusion.

First, an overview of PPRL systems is given, with particular emphasis on a thorough analysis of the most commonly used encoding techniques. Next, the existing GMA is introduced and explained in order to provide the basis for the study. In addition, an overview of ANNs is given to provide the necessary background knowledge.

Next, a detailed description of the attack model for the DEA is outlined, including how ANNs are used to enhance the attack. This is followed by an explanation of the actual implementation of the DEA, along with a discussion of the experiments conducted. The results of the DEA on different encoding schemes are then analysed and evaluated. Finally, this thesis concludes with a summary of the main contributions, a discussion of the broader implications, and suggestions for future research.

2. Background

This chapter provides an overview of the key concepts relevant to this thesis, including PPRL, various encoding techniques used in secure linkage and existing attacks on encodings and PPRL schemes. PPRL enables different organizations to link records belonging to the same individual across datasets while preserving privacy, making it a crucial tool. However, the security of such methods can depend on the encoding techniques used to transform sensitive data into a more privacy-preserving format [VCRS20].

To understand the vulnerabilities of PPRL systems, we examine three key encoding techniques: BF, TMH, and TSH, each offering different trade-offs between efficiency, privacy, and robustness against attacks. While these methods aim to prevent direct access to plaintext identifiers, they remain susceptible to adversarial techniques designed to infer or reconstruct the original data [SAH24; VCRS20].

One such adversarial approach is the GMA, which leverages structural similarities between encoded and non-encoded datasets to re-identify individuals. Although GMAs are good in reconstructing intersections of datasets, they are limited to the intersection of the two datasets. To extend GMAs beyond known intersections, we explore ANNs, which can learn complex patterns in encoded data and increase identification rates [SAH24; VCRS20].

By integrating these concepts, this chapter establishes the necessary foundation for understanding the DEA introduced later in this thesis, demonstrating how machine learning techniques can be employed to bypass existing privacy-preserving mechanisms.

2.1. Overview of PPRL

PPRL enables the linkage of records from different databases that refer to the same individual while trying to preserve privacy. Traditional record linkage relies on unique identifiers, but these are often unavailable or inconsistent due to variations in formatting, spelling or missing data in a distributed environment. Linking records directly on plaintext data poses significant privacy risks. To mitigate these risks and prevent further threats to individuals, regulations such as the European Union's General Data Protection Regulation have been established to govern the handling of PII [SAH24; VCRS20].

To enable record linkage using PII while trying to preserve privacy, PPRL employs similarity-preserving encoding on quasi-identifiers, allowing linkage to be performed on encoded representations rather than raw data. This approach protects identities while still facilitating record matching based on encoded similarities and probabilistic approaches. But this also causes the security of PPRL schemes to be dependent on the encoding techniques used [SAH24; VCRS20].

Broadly, PPRL methods fall into two categories, perturbation based techniques and Secure Multi-Party Computation (SMC) based techniques. SMC-based techniques provide strong security guarantees and high accuracy but suffer from computational and communication overheads. Conversely, perturbation-based techniques balance linkage quality, scalability, and privacy protection, making them more practical for real-world applications [VCRS20].

As seen in Figure 2.1 a typical PPRL system involves three main parties working together in four steps to enable the linkage while maintaining privacy. First data owners, who are responsible for maintaining their respective databases, D and D', encode the quasi-identifiers by agreeing on an encoding scheme with the corresponding parameters before then as a second step sharing their respective data sets with the linkage unit. The linkage unit, a trusted entity, performs the actual record linkage using only the encoded representations, without access to the original identifiers. Once the linkage is completed, the linkage unit assigns unique pseudonyms to the successfully linked records and returns the pseudonymized dataset as a third step to the data owners. The data owners then replace the linkage data with these pseudonyms before sending the dataset in the fourth and final step to the receipient requesting the data. In the case of a research project this could be an data analyst. The data analyst can merge the records based on the identifiers and proceed with further research and analysis, all while minimizing the risk of re-identification and protecting individual privacy [SAH24].

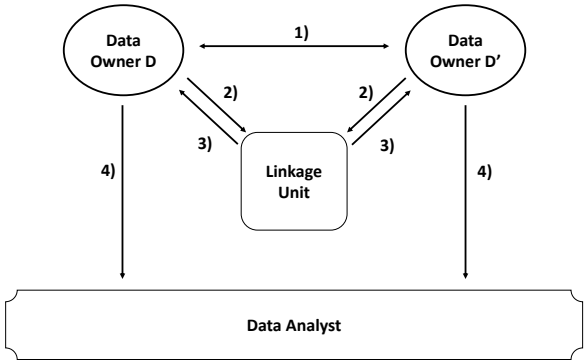

Figure 2.1.: Overview of the PPRL process.

Based on this scheme, each database record can therefore during the linkage process be represented as $r = (\lambda, \sigma)$, where λ denotes the linkage data which are encoded quasi-identifiers such as names and birthdates and σ refers to the remaining microdata like in a health care scenario patient information [SAH24].

Linkage in PPRL is performed probabilistically, meaning that two records, r and r', are considered linked if their similarity score on $sim(\lambda_r, \lambda_{r'})$ exceeds a predefined threshold. The choice of threshold plays a crucial role in balancing the quality of the linkage. Lower thresholds tolerate more variation and matches between records but increase the likelihood of false positives, while higher thresholds reduce false positives but may miss legitimate matches. Thus, selecting the optimal threshold is a trade-off that requires careful consideration of the specific goals and constraints of the linkage process [SAH24].

A robust PPRL scheme must satisfy several important criteria to ensure effective and secure linkage. First, a similarity function, $sim(\lambda_r, \lambda_{r'})$, must exist to determine if two records belong to the same entity based on a predefined threshold. Second, an encoding scheme $enc(\lambda)$ must be applied to λ in such a way that the linkage unit cannot reconstruct the original data. Finally, a function $sim(enc(\lambda_r), enc(\lambda_{r'}))$ must be available that allows similarity computations on encoded data. These requirements ensure both the privacy and the effectiveness of the record

linkage process [SAH24].

One common approach for measuring similarity for two quasi identifiers and enabling probabilistic matching on quasi identifieres is using n-grams. Here, string values are divided into overlapping substrings of length n using a sliding window approach [SAH24]. For example, for the string "encoding" with $n = 2$, the n-grams are:

$$\{\text{en}, \text{nc}, \text{co}, \text{od}, \text{di}, \text{in}, \text{ng}\}$$

The similarity of two sets of n-grams can then be computed using metrics such as the Dice Coefficient where X and Y denotes two sets which are in the case of PPRL the n-grams for two different pseudo identifiers [SAH24].

$$Dice(X, Y) = \frac{2 \cdot |X \cap Y|}{|X| + |Y|}$$

The Jaccard Similarity can be used in a similar way [SAH24].

$$Jaccard(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$$

PPRL has been successfully applied in various domains, demonstrating its practical importance in securely linking records across institutions while trying to preserve privacy. Notable applications include the Social Investment Data Resources, the Lumos Initiative, the Swiss National Cohort, and the Gemeinsamer Bundesausschuss. These implementations highlight the versatility of PPRL in diverse contexts, where privacy protection is essential while enabling effective data linkage for research and analysis [SAH24].

2.2. Key Encoding Techniques

In the context of PPRL, three primary encoding techniques have emerged: BF, TMH and TSH. These encoding methods are essential for transforming sets of quasi-identifiers into representations that preserve similarity information while trying to maintain privacy [SAH24; SBR09; VCRS20].

PPRL typically involves encoding sets of quasi-identifiers before linkage. A set in this context is generally a collection of n-grams, which are substrings of length n extracted from one or multiple attributes using a sliding window approach. Since input data varies in length, all encoding techniques in PPRL must take arbitrarily long inputs (sets of n-grams) and produce encoded outputs. This ensures that similarity computations can be efficiently performed on encoded data by the linkage unit without accessing the raw identifiers [SAH24; VCRS20].

Before linkage, data owners must agree on the encoding scheme and share necessary cryptographic secrets to facilitate secure comparison. Among the available techniques, BFs are the most widely used approach for record linkage [SAH24].

2.2.1. BF

BFs were originally developed for efficient membership testing in set structures without requiring direct access to the sets themselves [Blo70]. Due to their ability to efficiently compute set similarities in a privacy-preserving manner, they have been widely adopted in PPRL applications [SAH24; SBR09; VCRS20].

A BF $b \in \{0,1\}^l$ is a bit vector of length l. It uses $k \geq 1$ independent hash functions $H = \{h_1, h_2, ..., h_k\}$, where each function maps an arbitrary input to a position in the filter [SAH24; SBR09]:

$$h_i : \{0,1\}^* \rightarrow \{1, ..., l\}, \quad \forall i \in \{1, ..., k\} \tag{2.1}$$

Initially, the BF is set to all zeros. Each element $s \in S$ is hashed with every function h_i, and the corresponding bit positions in the filter are set to 1 [SAH24; SBR09]:

$$\forall s \in S, \forall h_i \in H, \quad b[h_i(s)] = 1 \tag{2.2}$$

An Example for this can be seen in Figure 2.2 where the set of 2-grams for the word "encoding" is encoded using a BF with $k = 2$ hash functions. As can be seen, the 2-grams are hashed using the two hash functions and the corresponding bits are set to 1 in the BF.

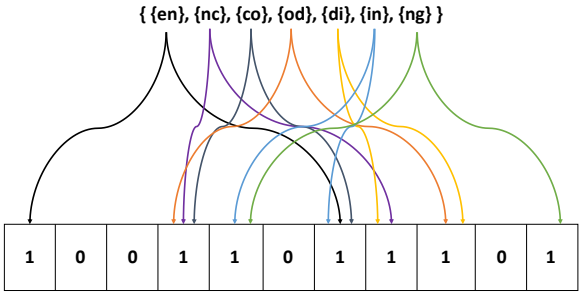

Figure 2.2.: Example BF for $k = 2$ hash functions on the set of 2-grams for "encoding".

Since BFs are binary vectors, the similarity between two BFs, b_1 and b_2, is computed based on the overlapping 1-bits based on their position. The Dice Coefficient is commonly used for this purpose [SAH24], providing a measure of similarity by comparing the overlap of 1-bits in the two binary vectors. Therefore encodings using BF encoding allows for efficient computation of similarity between two sets which is benefical for PPRL systems.

$$\text{Dice}(b_1, b_2) = \frac{2 \cdot |b_1 \cap b_2|}{|b_1| + |b_2|} \tag{2.3}$$

Because the sets in a PPRL systems consists of n-grams, a deterministic relationship between the n-grams present in the quasi-identifier λ and the set bits in the BF is created. However, due to the finite length of BFs, collisions occur where different n-grams map to the same bit position (see Figure 2.2). While this can cause incorrect linkages, it also enhances privacy by distorting frequency distributions [SAH24; VCRS20].

Three primary approaches exist for applying BFs to sensitive data. The first approach, Attribute-Level BFs, encodes each attribute, such as first name or last name, into a separate BF, enabling multiple similarity computations. However, Attribute-Level BFs are more vulnerable to frequency-based privacy attacks as they lower the collisions which would occur using only one BF. The second approach, Cryptographic Long-Term Key Encoding, merges

multiple attributes into a single BF, reducing vulnerability to frequency attacks but remaining susceptible to pattern-mining-based attacks. Finally, Record-Level BFs employ a weighted bit sampling technique to minimize frequency information, enhancing privacy protection while maintaining high linkage quality. Each of these approaches balances privacy concerns with the need for accurate and effective record linkage [VCRS20].

Several privacy-enhancing methods have been proposed to mitigate frequency attacks in BFs. These techniques introduce a trade-off between privacy and linkage quality. One such method is balancing, which ensures an equal number of 1-bits across BFs, thereby reducing the likelihood of frequency-based attacks. Another approach is salting, which randomizes bit positions to prevent direct inference from the encoded quasi-identifiers. Additionally, XOR folding is used to reduce the BF length while maintaining the bit-wise dependencies necessary for effective linkage. These methods aim to strengthen privacy while retaining the accuracy of the linkage process [SAH24; VCRS20].

A major improvement was introduced by Armknecht et al. [AHS23], who proposed a diffusion layer for BF encodings. This method generates Encoded Linkage Data (ELD), where each bit is computed as the XOR sum of multiple BF bits. The indices for XOR computations are randomly chosen and secretly shared among data owners [AHS23].

By applying diffusion, the deterministic relationship between 1-bits in the ELD and the original n-grams is broken, improving privacy while still enabling approximate matching [AHS23].

BFs remain a core technique in PPRL due to their efficiency and scalability. However, their vulnerability to frequency attacks has led to improvements such as Record-Level BFs and diffusion layers, which enhance privacy at the cost of increased computational complexity [AHS23; SAH24; VCRS20].

2.2.2. TMH

TMH is a variation of MinHash initially introduced for efficient estimation of set similarities and later adapted for privacy-preserving probabilistic record linkage. MinHash itself was first proposed in the context of document resemblance and containment estimation. TMH extends MinHash by employing tabulation-based hashing, which enhances its security compared to BFs [Bro97; VCRS20].

MinHash aims to approximate the Jaccard similarity between two sets, S and S'. The fundamental idea is to represent both sets as sequences of randomly ordered elements and apply multiple rounds of random permutations π to shuffle them. After each round, the first elements of both sequences are compared. The larger the intersection between S and S', the higher the probability that the first elements will match. An example for this can be seen in Figure 2.3 where the sets are permutated twice and the first element is compared for each new set. The final Jaccard similarity estimate is computed based on the number of collisions achieved during permutation, in the example is one collision for two permutations [Bro97; SAH24; VCRS20].

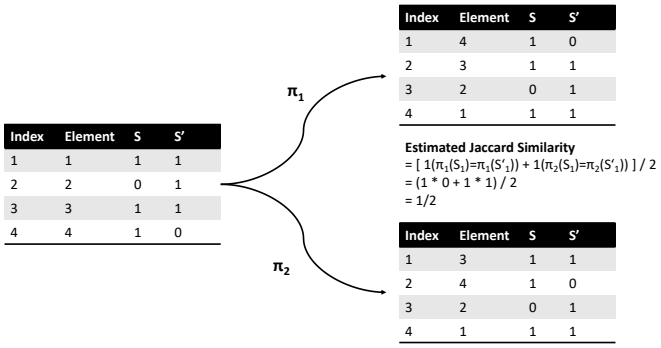

Figure 2.3.: Example computing approximate Jaccard similarity using MinHash with $\pi = 2$ permutations.

Instead of explicitly computing these permutations, MinHash simulates them by applying a suitable hash function to the elements of a set and selecting the smallest hash value as the representative signature. This is equivalent to sorting set elements by their hash values and returning the first element [SAH24].

Tabulation-based hashing is a technique used in TMH that provides efficient and high-quality hash functions by leveraging precomputed lookup tables. This method operates as follows [VCRS20]:

The process begins with the **initialization** of l sets of lookup tables, each containing c tables. Each table holds randomly generated bit strings for keys of length k, with a key space of 2^k. During the **hashing process**, each element in S is hashed using a one-way hash function, producing a fixed-length binary value. This binary value is then split into c sub-keys, each of length k. Each sub-key is used as an index to retrieve a random bit string from the corresponding lookup table, and the retrieved c bit strings are XORed together to produce a single output bit string. In the **MinHash signature generation** phase, this process is repeated for each of the l lookup table sets, and the minimum value among all generated bit strings is selected as the MinHash signature [SAH24; VCRS20].

An example for this can be seen in Figure 2.4 where the first element of the set is hashed using the first lookup table and the fourth hash function. The key is split into $c = 4$ sub-keys of length $k = 2$ and used to retrieve the corresponding bit strings from the lookup table. The retrieved bit strings are then XORed together to produce the final value for the first element.

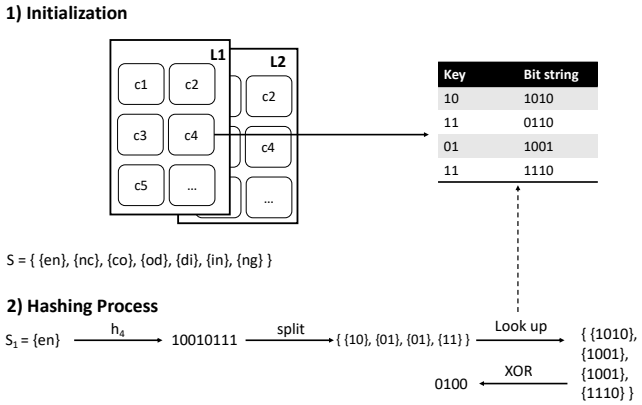

Figure 2.4.: Simplified TMH hasing step for the first lookup table, fourth hash function on the first set element.

To further enhance privacy, TMH employs a 1-bit hashing mechanism, where only the least significant bit of each MinHash signature is retained. These l bits are then concatenated to form the final bit array used as an encoded representation [SAH24].

The main advantage of TMH over BFs is its improved resistance to frequency-based attacks due to the complexity introduced by tabulation-based hashing. However, this security enhancement comes at a cost. It leads to higher computational overhead, as the need to generate and access multiple lookup tables increases processing time. Additionally, there is increased memory consumption because storing large precomputed tables requires additional space. These trade-offs must be considered when choosing between TMH and other privacy-preserving techniques [SAH24; VCRS20]. Despite these trade-offs, TMH remains an attractive alternative for PPRL due to its robustness against adversarial attacks.

Similar to BFs, TMH encodes each record as a bit vector of length l. Given two TMH-encoded bit vectors, their similarity can be estimated using a modified Jaccard coefficient, adapted to account for artificial bit collisions caused by truncation to the least significant bit. The Jaccard coefficient can also be converted into the Dice coefficient for improved comparability with BF-based methods [SAH24; VCRS20].

Overall, TMH provides a more secure encoding alternative to BFs in PPRL, though at the expense of increased computational and memory requirements [SAH24; VCRS20].

2.2.3. TSH

TSH is the most recent encoding scheme proposed for PPRL, introduced in 2020 [RCS20]. TSH was designed to address both the privacy vulnerabilities of BFs and the computational complexity of TMH while maintaining accuracy in similarity calculations. Similar to other encoding techniques, TSH requires the input to be split into a set of n-grams S prior to encoding [RCS20].

As a result of TSH encoding, each record from a sensitive database is represented by a set of integers, which can be directly used to compute Jaccard similarity. TSH employs two distinct hashing steps. In the first hashing step, the input set is converted into a bit matrix representation. In the second hashing step, the bit matrix columns are mapped into integers, enabling efficient comparison. This two-step process allows TSH to represent sensitive data

in a way that facilitates effective similarity computation while preserving privacy [SAH24]. These steps provide accurate Jaccard similarity calculations while improving privacy protection compared to traditional BF-based encodings [RCS20].

In the first step of the TSH process, elements of the n-gram set S are hashed into k independent BFs b_i of length l, meaning each hash function results in a corresponding bit vector. This generates a $k \times l$ matrix, where each row corresponds to a BF created using a unique hash function, and each column represents the bitwise state across all BFs for a given position. In the second step, after constructing the bit matrix, TSH computes column-wise hashes to convert the bit vectors into integer representations . Each column vector is treated as an input for a hash function, and all-zero columns are skipped to prevent distortion in similarity calculations since they do not encode any n-grams. To enhance security and avoid hash collisions between columns with identical bit patterns, a salt value and the column index are concatenated before hashing [RCS20].

The final integer representation for each column i for $1 \leq i \leq l$. is computed as [RCS20; SAH24]:

$$H(salt, i, b_{1i}, b_{2i}, ..., b_{ki}) \tag{2.4}$$

The output of the second hashing step is a set of integers, allowing similarity computations using the Dice coefficient rather than directly computing bitwise similarity. Since the encoded data consists of sets, Jaccard similarity can also be computed similarly to MinHash-based encodings [RCS20].

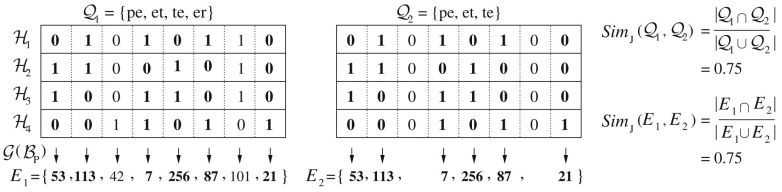

Figure 2.5.: TSH example for for two input values "peter" and "pete" [RCS20].

An illustrative example is provided in Figure 2.5, where the set of 2-grams for the words "*peter*" and "*pete*" is encoded using TSH with $k = 4$ hash functions and a Bloom filter length of $l = 8$. In the first hashing step, each 2-gram is processed using the k hash functions, resulting in a 4×8 bit matrix. Each row of this matrix corresponds to a Bloom filter generated with a distinct hash function, encoding the presence of 2-grams across the l bit positions.

In the second hashing step, the columns of the bit matrix are transformed into integer values. This is achieved by applying an additional hash function that incorporates both a salt value and the column index, effectively compressing the binary representation into a fixed-length integer vector. The final output is a set of integers representing the encoded word, which can subsequently be used to compute similarity scores between different words [RCS20].

To improve efficiency, TSH can be implemented using a Pseudo-Random Number Generator (PRNG) instead of cryptographic hash functions. The PRNG is seeded with the value to be hashed before generating random numbers, ensuring that the sequence of generated values depends deterministically on the input [RCS20].

By combining efficient bit vector representations with integer-based similarity computations, TSH offers a balance between privacy, security, and computational efficiency, making it a promising alternative to existing PPRL encoding schemes [RCS20; VCRS20].

2.3. GMA

GMAs were introduced by Vidanage et al. [VCRS20] and represent the most significant threat to PPRL due to their universal applicability. Unlike traditional cryptanalytic attacks, GMAs exploit the fundamental properties of non-interactive PPRL to compromise the security of all schemes relying on similarity-preserving encoding [SAH24].

Non-interactive PPRL refers to linkage schemes where data owners independently encode their data and share it with a linkage unit. The linkage unit then performs record matching solely based on the encoded data, without requiring further interaction with the data owners during the linkage process. This approach minimizes communication overhead and computational complexity, as no iterative exchanges between parties are necessary. In contrast, interactive PPRL methods involve multiple rounds of communication between data owners and the linkage unit to refine matching results or improve accuracy [KKM+14].

In PPRL, encoded records are linked based on similarity computations. Since these similarities serve as identifiers, an attacker with access to an encoded dataset can leverage an auxiliary plaintext dataset to re-identify individuals. The latest version of the GMA developed by Schaefer et al. [SAH24] overcomes the limitations of the original attack by Vidanage et al. [VCRS20] and enhances success rate and robustness, even under limited knowledge scenarios [SAH24].

In the context of a GMA on a PPRL system, the attacker is modeled as the linkage unit and is assumed to have minimal prior knowledge. The attacker does not know any encoding secrets, seeds, or salts used in the system to protect the data. The only information available to the attacker is that which is inevitably known to the linkage unit during the linkage process. This assumption ensures that the attacker can only exploit data accessible through normal system operations, adhering to Kerckhoffs's principle [SAH24].

Since the attack does not depend on specific encoding parameters or attribute frequency distributions, it is universally applicable as long as pairwise similarities of encoded data are available [SAH24].

The first step of the attack involves constructing similarity graphs for both the encoded dataset (D_{enc}) and the plaintext dataset (D_{plain}). In these graphs, each node represents an individual record, while edges between nodes are assigned weights based on pairwise similarity computations. To ensure computational efficiency and focus only on meaningful connections, edges with similarity scores below a predefined threshold are omitted, reducing noise and improving the accuracy of the attack [SAH24].

Since certain encoding properties, such as BF length (l), are inevitably known to the linkage unit, D_{plain} can be transformed analogously to D_{enc} for effective comparison. Importantly, this step does not require knowledge of shared secrets, as the primary objective is to replicate the effect of encoding on similarity [SAH24].

To quantify the structural similarity between nodes in G_{plain} and G_{enc}, node embeddings are computed to transform the graph structure into a numerical representation. This process begins with graph embedding using the Node2Vec algorithm, which applies a Word2Vec-like approach to learn vector representations of nodes. During this process, nodes undergo multiple random walks, where each walk simulates a sequence of transitions between connected nodes. These sequences are then treated as sentences, allowing the model to learn embeddings that capture the local and global structure of the graph. The behavior of these random walks is controlled by two hyperparameters: p, which determines the likelihood of returning to a previously visited node, and q, which influences the tendency to explore new regions of the graph. The result is an embedding matrix where each row represents a node as a vector in Euclidean space [SAH24].

Once embeddings are generated, they must be aligned to allow meaningful comparison between the two graphs. Due to the randomness inherent in embedding generation, direct comparison is not possible. Instead, an iterative approach is used to solve two subproblems: first, an optimal linear transformation is determined using Procrustes Analysis to align the embeddings, and second, node correspondences are established via the Sinkhorn Algorithm, which minimizes the Wasserstein distance between the distributions of embeddings in both graphs. To achieve an effective alignment, an unsupervised stochastic optimization scheme alternates between these two steps over n epochs, gradually refining the transformation and correspondences until convergence [SAH24].

Once the embeddings from the plaintext and encoded datasets are aligned, the reidentification process can begin. Each embedding in the transformed plaintext space is compared to its counterparts in the encoded space, with similarity measured using cosine similarity. This metric quantifies how closely two embeddings align in the high-dimensional space, enabling the attacker to identify records in the encoded dataset that most closely resemble those in the plaintext dataset [SAH24].

The final step involves constructing a bipartite graph, where nodes from the plaintext and encoded datasets are linked based on their similarity scores. To determine the optimal mapping, the Jonker-Volgenant algorithm is applied, ensuring that each node in the smaller dataset is uniquely matched to a corresponding node in the larger dataset. This algorithm maximizes the total similarity across all matched pairs, effectively revealing the identities of individuals within the encoded dataset [SAH24].

An example of this process is illustrated in Figures 2.6 and 2.7, which provide a high-level overview of the GMA attack applied to a PPRL approach using BF encoding. Initially, the two data owners agree on an encoding scheme and encode their respective datasets, for this example using BF. These encoded datasets are then send to the linkage unit. The attack begins at this point, leveraging information inherently available to the linkage unit. Specifically, the linkage unit constructs similarity graphs for both the encoded datasets and an auxiliary plaintext dataset. By embedding these similarity graphs into a vector space and aligning the embeddings, the attacker can identify re-identifications by comparing embeddings and constructing a bipartite graph that matches entries from the encoded dataset to those in the plaintext auxiliary dataset [SAH24].

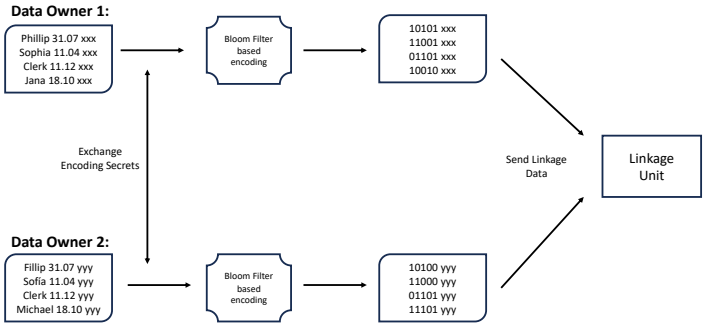

Figure 2.6.: High-level overview of the GMA attack process. Two data owners encode their datasets and send them to the linkage unit.

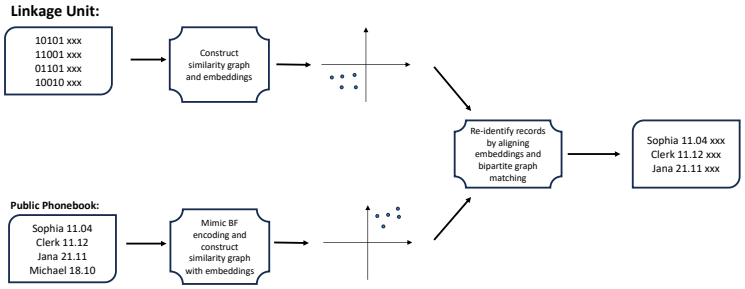

Figure 2.7.: High-level overview of the GMA attack process. The linkage unit mimics the BF encoding for the public dataset and creates for both datasets similartie graphs, embeddings and aligns them to perform bipartite matching.

The improved GMA approach by Schaefer et al. [SAH24] achieves near-perfect re-identification rates when dataset overlap is 100%. Even for low-overlap scenarios (e.g., 5%), success rates reach 99.9% for TSH [SAH24]. The only encoding scheme resistant to GMAs is BFs with diffusion layers, which disrupts similarity preservation for sufficiently high diffusion values [SAH24].

2.4. ANN

ANNs are a class of machine learning models inspired by the structure and function of biological neural systems. They consist of interconnected layers of artificial neurons that process input data and extract meaningful patterns through iterative learning. ANNs have been widely applied in various fields, including image recognition, natural language processing, and classification tasks. Neural networks are particularly effective for complex tasks because they

automatically identify and refine patterns in data through multiple layers of processing, removing the need for manual feature engineering, which can be difficult and time-consuming [DKK+12].

The structure of an ANN consists of multiple layers as can be seen in Figure 2.8, each serving a distinct role in processing and transforming input data. The input layer is the first stage of the network, responsible for receiving raw data and forwarding it to subsequent layers. The number of neurons in this layer corresponds directly to the number of input features, ensuring that all relevant information is passed through the network [DKK+12; SMNP24].

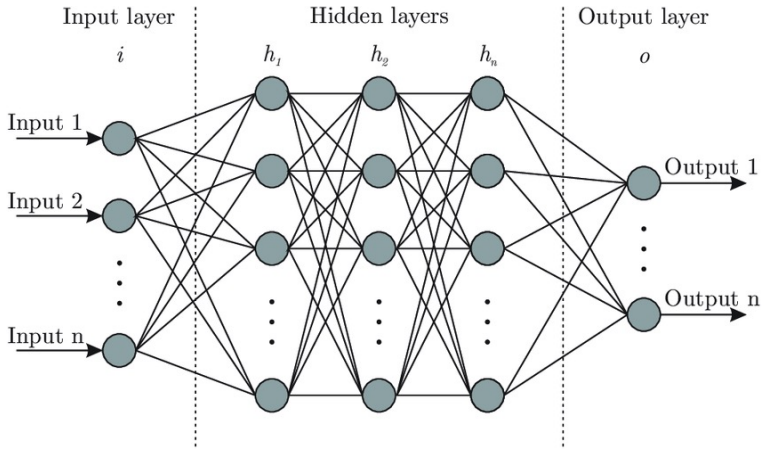

Figure 2.8.: ANN consisting of multiple input neurons (input layer), hidden layers and output neurons (output layer) [BGF17].

Following the input layer are the hidden layers, which perform feature extraction and transformation. Each neuron in a layer applies a weighted sum operation to its inputs, followed by an activation function that introduces non-linearity, enabling the network to learn complex patterns in the data. Common activation functions include the Rectified Linear Unit (ReLU), Sigmoid, Leaky ReLU, and Exponential Linear Unit (ELU), each offering advantages depending on the specific task [RN16; SSA17].

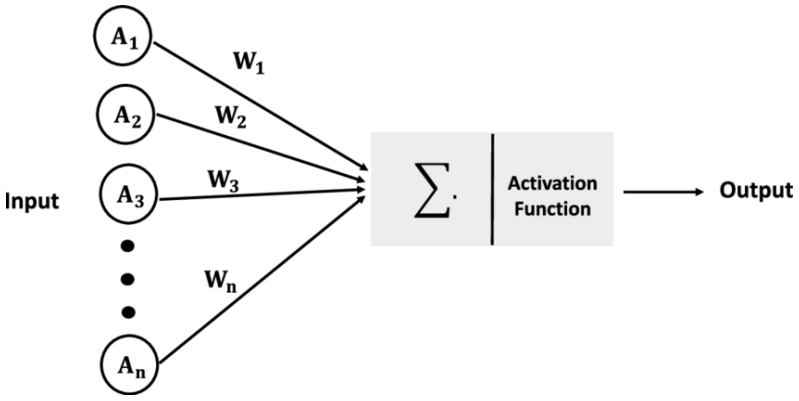

Figure 2.9.: Sketch of an single artifical neuron in an ANN [GSB+21].

An example of this process is illustrated in Figure 2.9, which depicts a single neuron in an ANN. The neuron takes multiple inputs, multiplies them by corresponding weights, sums the weighted inputs along with a bias term, and applies an activation function to compute the final output. Mathematically, this operation is represented as

$$a = \sigma(z) = \sigma\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{2.5}$$

where a is the neuron's output, σ is the activation function, z is the weighted sum, w_i are the weights, x_i are the inputs, and b is the bias term [GSB+21]. The depth and size of the hidden layers determine the network's capacity to model relationships, making them a crucial component of deep learning architectures [DKK+12; SMNP24].

Finally, the output layer generates the final predictions based on the processed information. The number of neurons in this layer depends on the nature of the task, whether it is a classification problem, where each neuron represents a class, or a regression task, where a single neuron outputs a continuous value [DKK+12].

Training an ANN involves iteratively adjusting its parameters, using a labeled dataset to minimize prediction errors. The process begins with forward propagation, where input data flows through the network, passing through multiple layers until it reaches the output layer, generating a prediction. This prediction is then compared to the actual target value, and the discrepancy between the two is quantified using a loss function, which measures the model's performance [RN16].

To improve accuracy, the network undergoes backward propagation (backpropagation), where the gradient of the loss with respect to each weight is computed using the chain rule of differentiation. These gradients indicate how each parameter should be adjusted to reduce the overall error [RN16].

An optimizer, such as Stochastic Gradient Descent (SGD) or Adam, updates the weights accordingly by taking small steps in the direction that minimizes the loss. The training process is repeated over multiple epochs, where the entire dataset is processed multiple times. To enhance efficiency, the data is often divided into batches, allowing the model to update its weights incrementally rather than processing the entire dataset at once. Over time, this iterative optimization process enables the network to learn meaningful patterns and improve its predictive performance. ANNs can be applied to different classification tasks, depending on whether a data instance belongs to a single category or multiple categories simultaneously [RN16].

In traditional single-label or binary classification problems, each instance is assigned to one and only one category from a predefined set of classes. To achieve this, the network's output layer typically uses a Softmax activation function, which converts the raw output scores into a probability distribution over all possible classes. The model can be trained using the Cross-Entropy Loss function, which penalizes incorrect classifications by measuring the difference between the predicted probability distribution and the actual class label [HCR+16; RN16].

In contrast, multi-label classification allows an instance to belong to multiple categories at the same time. Instead of a single categorical output, the network produces independent predictions for each possible label. The output layer can then as an example apply sigmoid activations for each label, transforming the raw scores into independent probabilities indicating the presence or absence of each class. Since each label is treated as a separate binary classification problem, Binary Cross-Entropy (BCE) Loss is commonly used to optimize the

model, ensuring accurate predictions across multiple labels [HCR+16; RN16].

Different ANN architectures have been developed to address various problem domains, each optimized for specific types of data and tasks. Feedforward ANNs (FNNs) represent the simplest architecture, where data flows in one direction from the input layer to the output layer without forming cycles. These networks are widely used for basic classification and regression tasks but may struggle with complex patterns that require spatial or sequential dependencies [GB10; RN16].

For tasks involving image processing, Convolutional ANNs (CNNs) are commonly used. CNNs employ convolutional layers that apply filters to input images, allowing the network to capture complex patterns such as edges, textures, and shapes. This makes them highly effective for applications like object recognition and medical imaging [ON15; SMNP24].

When dealing with sequential data, Recurrent ANNs (RNNs) and their advanced variant, Long Short-Term Memory (LSTM) Networks, are particularly useful. These architectures introduce recurrent connections, enabling them to maintain memory of previous inputs and recognize patterns over time. This makes them well-suited for natural language processing, speech recognition, and time series forecasting [MJ+01].

2.4.1. PyTorch for Training ANN

Training ANNs in PyTorch involves defining and optimizing a model through iterative learning, leveraging GPU acceleration for efficient computation. PyTorch offers a flexible and modular framework for designing and training neural networks, making it a widely adopted tool in deep learning research and applications [Fou].

The process begins with defining the model architecture using the ‘torch.nn.Module‘ class, where users specify layers, activation functions, and parameters. This approach provides full control over the forward propagation process. Additionally, a task appropriate loss function is selected [Fou].

After defining the model and loss function, an optimizer is chosen to adjust the model's weights during training. Training is conducted over multiple epochs, during which the dataset is processed in successive iterations to improve model accuracy. To enhance computational efficiency and stabilize gradient updates, the dataset is divided into mini-batches, enabling partial processing instead of loading the full dataset into memory [Fou].

One of PyTorch's key advantages is its support for GPU acceleration via ‘torch.cuda‘, which significantly reduces training time for large datasets. By moving tensors and models to a GPU, computations are performed in parallel, leading to substantial performance gains compared to CPU-based training. Additionally, PyTorch's autograd engine enables automatic differentiation, simplifying the backpropagation process and making model optimization more efficient [Fou].

Despite their success, ANNs present several challenges that must be carefully managed to ensure robust and efficient learning. One of the most common issues is overfitting, where a model becomes too specialized in learning patterns from the training data, capturing possible noise rather than generalizable features. This leads to poor performance on unseen data. Techniques such as dropout regularization, L2 weight decay, and early stopping are commonly used to mitigate overfitting and improve generalization [Fou; GB10].

Dropout regularization is a technique used to prevent overfitting by randomly setting a fraction of neurons to zero during training. This forces the network to learn more robust features by preventing it to rely too heavily on a single neuron to perform well. L2 weight

decay is another regularization method that penalizes large weights by adding a regularization term to the loss function. It tries to prevent the network from applying too much importance to a single feature, encouraging it to learn more generalizable patterns. Early stopping is a simple yet effective technique that stops training when the model's performance on a validation set starts to degrade below a certain threshhold, preventing overfitting [Fou; GB10].

Another fundamental challenge is the vanishing and exploding gradient problem, which occurs in deep networks during backpropagation. When gradients become too small (vanishing), weight updates diminish, leading to slow or stalled learning. This can happen because gradients are the product of multiple derivatives, which can cause them to shrink exponentially as they propagate through the network. This can happen especially using activiation functions that saturate for extreme values. Conversely, when gradients grow too large (exploding), unstable updates cause erratic training behavior. Solutions such as batch normalization, gradient clipping, and advanced activation functions like Leaky ReLU help address these issues [Fou; GB10].

Batch normalization stabilizes training by normalizing the inputs to each layer, ensuring that activation values remain within a consistent range. Gradient clipping addresses the problem of exploding gradients by imposing a threshold on gradient values, thereby maintaining training stability. Leaky ReLU, an activation function, mitigates the vanishing gradient problem by allowing a small, non-zero gradient for negative inputs, enabling continued learning even with extreme input values [Fou; GB10].

The computational complexity of deep learning models is another major concern, as large-scale ANNs require extensive memory and processing power. Training deep networks on large datasets can be prohibitively slow on CPUs, necessitating the use of GPUs or specialized hardware like TPUs (Tensor Processing Units) to accelerate training. Efficient data-loading techniques and mixed-precision training can further optimize computational efficiency. PyTorch leverages this by utilizing its ‘torch.utils.data.DataLoader‘ class to efficiently load and preprocess data by using multiple workers. Mixed-precision training is a technique that partly uses lower-precision floating-point numbers to reduce memory usage and speed up computations, while still maintaining model accuracy [Fou].

Lastly, hyperparameter tuning plays a critical role in model performance. Selecting the right learning rate, batch size, number of layers, and optimization algorithm requires experimentation and fine-tuning. Automated methods such as grid search, random search, and Bayesian optimization can assist in finding optimal configurations, but these processes are computationally expensive. Addressing these challenges effectively is crucial for developing high-performing ANNs that generalize well across different datasets and tasks [Fou].

3. Methodology

The DEA is a novel attack method that extends the capabilities of GMAs by moving beyond the intersection of datasets to re-identify individuals who were previously unmapped. This chapter outlines the methodology behind the DEA, including modifications to the GMA, the design and implementation of the DEA itself, and the use of ANNs to enable probabilistic reconstruction of PII from encoded data.

The DEA builds upon the GMA by using its re-identification results as a foundation for further inference. While the GMA re-identifies only those records that exist in both the attacker's dataset and the encoded target dataset, it can leave a substantial portion of records unmapped. The goal of the DEA is to extend this re-identification process by applying a machine learning–based approach to infer the missing PII of the remaining records using a trained ANN.

To achieve this, the DEA follows a structured pipeline comprising six key steps, as illustrated in Figure 3.1. The first step involves executing the GMA and extracting its results in a predefined format to serve as training data. This dataset includes the re-identified individuals, their corresponding encoded representations, and the plaintext information that was successfully linked. In addition, the GMA results contain the non-re-identified individuals, who are represented solely by their encoded PII and associated plaintext values.

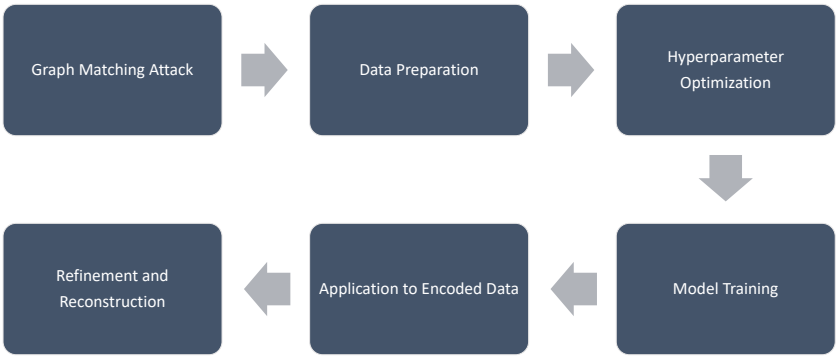

Figure 3.1.: Overview of the DEA attack pipeline.

Once the data is extracted, it undergoes a transformation process to prepare it for ANN training. This data preparation step involves constructing specialized datasets that convert the encoded representations and their corresponding labels, plaintext n-grams, into tensor-based formats suitable for processing by deep learning models. The resulting datasets are subsequently split into training, validation, and test subsets, and corresponding data loaders are created to facilitate efficient mini-batch processing during model training.

With the data pipeline established, hyperparameter optimization is performed to determine the most effective model configuration. This process systematically explores combinations of hyperparameters, such as the number of hidden layers, hidden layer size, activation functions, optimizers, and learning rate schedulers, to identify those that yield the best performance. Hyperparameter tuning is essential because different encoding schemes with different parameters necessitate tailored model configurations to effectively capture the underlying patterns in the data. The selected hyperparameters are then used to define the ANN architecture, which is trained to learn the mapping between encoded representations and their corresponding plaintext n-grams.

The architecture of the ANN is tailored to the specific characteristics of the encoding schemes used in the PPRL scheme. The input layer size is determined by the dimensionality of the encoded representation, while the output layer size corresponds to the size of a predefined n-gram dictionary.

Once the best hyperparameter configuration is identified, the ANN is trained using the re-identified individuals as labeled data. Training proceeds over multiple epochs, during which the model iteratively processes the training dataset, computes the loss, and updates its parameters via backpropagation. Performance is continuously monitored on the validation set to track generalization and prevent overfitting.

Once the ANN is trained, it can be applied to the set of non-re-identified individuals, i.e., records that remained unmapped after the GMA. This dataset serves as the test set during the experimentation phase to evaluate the performance of the attack.

The model outputs a probability distribution over an index that maps to a dictionary of n-grams, indicating the likelihood of each n-gram being present in the corresponding plaintext. To refine these predictions, a thresholding mechanism is applied to filter out low-confidence outputs and retain only the most probable n-grams. These predicted n-grams are then aggregated and reconstructed into potential PII, constituting the final step of the DEA process.

This methodological approach represents an advancement in attacking PPRL systems. By leveraging deep learning techniques, the DEA enables an attacker to infer sensitive personal information beyond the scope of traditional GMA approaches. The following sections provide a detailed discussion of each component, including the design choices, implementation details, and challenges encountered during development.

3.1. Problem Definition

The primary challenge that the DEA seeks to address is the limited scope of re-identifications achieved by the GMA. While the GMA effectively links records by exploiting structural relationships within encoded datasets, its success is inherently restricted to individuals who are present in both the plaintext and the encoded datasets and can be matched based on graph similarity. However, in real-world scenarios, there may exist additional re-identification potential beyond these direct matches.

One possible way to extend re-identifications is to rerun the GMA iteratively, incorporating additional publicly available data to gradually refine the matching process. However, this approach is dependent on the availability and quality of external data sources, which may not always be feasible. Instead, the DEA introduces a novel strategy that aims to reconstruct deterministic relationships between encoded representations and their corresponding plaintext information. This is based on the observation that all encoding schemes used in PPRL rely on

hash functions or other deterministic mappings.

Hash functions, for example, produce fixed-length outputs from inputs of arbitrary length and are deterministic, meaning the same input will always yield the same output. The DEA leverages this property by training ANNs to learn statistical relationships between encoded values and the original n-grams of PII. The objective is to recover the most probable plaintext representation given an encoded input, effectively framing the attack as a probabilistic, frequency-based inference problem. However, several challenges complicate this task.

The first major challenge is the lack of knowledge about the specific number and type of hash functions used during encoding. As a result, the model must learn patterns in the data without any explicit understanding of the underlying hashing mechanisms. Fortunately, this limitation is partially mitigated by the fact that the DEA does not rely on a one-to-one mapping between hash outputs and plaintext n-grams, but instead depends on statistical inference across large numbers of training samples.

A more fundamental challenge arises from the collision property of hash functions. Because hash functions map an infinite input space to a finite output space, different inputs may produce identical hash values, making it inherently difficult to perfectly recover the original plaintext. These collisions introduce uncertainty into the re-identification process, preventing the DEA from achieving perfect reconstruction accuracy. Consequently, the predictions made by the DEA are probabilistic rather than deterministic. Therefore it can estimate the likelihood of a specific n-gram being present in the original PII, but cannot guarantee absolute correctness.

The primary reason the GMA alone is unable to achieve perfect reconstruction is that it relies solely on structural similarities within the dataset, without attempting to infer direct relationships between encoded values and their plaintext equivalents. In contrast, the DEA enhances the capabilities of the GMA by enabling the reconstruction of individual plaintext components directly from encoded representations, thereby increasing the overall re-identification potential. This novel approach improves the effectiveness of the attack, allowing for the re-identification of individuals who were previously unmatchable using graph-based techniques.

3.2. Attacker Model

The attacker in the DEA scenario is modeled as the linkage unit within a PPRL protocol. This aligns with standard threat models in the literature, where the linkage unit is typically assumed to be semi-honest or honest-but-curious executing the prescribed protocol while remaining interested in extracting sensitive information from the encoded data it processes [SAH24].

Following Kerckhoffs's principle, the attacker is assumed to possess full knowledge of the PPRL system design, including encoding algorithms (e.g., BF construction), parameter settings (e.g., filter length, n-gram size, number of hash functions), and record linkage procedures. However, any encdoing specific parameters, such as secrets, random seeds, or salt values, are assumed to be unknown to the attacker.

In this setting, the attacker is presented with two encoded datasets originating from, for example, two organizations engaging in PPRL, and aims to re-identify individuals across them.

The attacker is assumed to operate in an offline setting without time constraints, allowing the use of exhaustive search, large-scale training, and iterative optimization. Given that linkage units are often embedded in national statistical agencies, health departments, or research consortia, it is realistic to assume access to computational resources, including parallel processing and GPU acceleration.

The primary goal of the attacker is to maximize the overall re-identification rate, demonstrating that individuals not re-identified through traditional GMAs can still be decoded using more advanced techniques. To assess the effectiveness of the DEA, its performance is compared against a baseline strategy that predicts, for each record, the k most frequent n-grams observed in the training data.

This baseline represents a naïve yet plausible strategy. A successful DEA must outperform this baseline to substantiate its threat to real-world PPRL deployments.

3.3. Modular Design of the DEA

The DEA aims to reconstruct plaintext PII from encoded records using machine learning techniques. A central challenge in implementing the DEA lies in the diversity of encoding schemes used to protect sensitive data. As each encoding method transforms plaintext into distinct numerical representations, the DEA must adapt both the dataset structure and the ANN architecture accordingly.

To address this, the DEA adopts a modular design: while the overall attack methodology remains consistent, specific implementations are tailored to each encoding scheme. Although the input representation and network architecture vary depending on the encoding, the output format is kept uniform across all models. The attack is framed as a multi-label classification task, where the ANN predicts the likelihood of individual n-grams appearing in the original PII. For each encoding scheme, a dedicated dataset structure transforms encoded records into a format suitable for ANN training. Moreover, a custom ANN architecture is employed for each encoding scheme to ensure the model effectively learns the mapping from encoded data to plaintext n-grams.

3.4. Step 1: GMA

3.4.1. Running the GMA

The GMA constitutes the first step in our DEA pipeline, establishing a foundation by identifying overlapping individuals between the encoded (target) dataset and the auxiliary (attacker) dataset. It exploits structural similarities between records to perform graph-based matching and re-identification of individuals.

In this phase, we apply the adjusted GMA implementation by Schaefer et al. [SAH24], which builds upon and extends the original approach introduced by Vidanage et al. [VCRS20]. This attack yields a partial mapping between records in the encoded dataset and plaintext identities from the auxiliary dataset. The successfully re-identified individuals represent the known intersection between both datasets and serve as labeled training data for the inference component of the DEA.

The output of this step comprises two distinct sets of records: (1) re-identified individuals, with known plaintext identities and their corresponding encoded representations, and (2) non-re-identified individuals, whose encoded representations remain unmapped. These unmapped encodings form the target set for the neural network-based reconstruction in the subsequent inference phase.

The effectiveness of the DEA is directly influenced by the quality of the GMA output. A higher re-identification rate in the GMA provides a larger training set for the ANN, improving

its ability to infer plaintext n-grams and reconstruct sensitive information for non-re-identified individuals. Conversely, a low re-identification rate limits the availability of labeled data and diminishes the overall reconstruction capability of the DEA.

3.4.2. Modifications to the GMA

To integrate the GMA as a preprocessing step for the DEA, modifications were made to the original implementation by Schaefer et al. [SAH24]. While the core algorithm remains unchanged, adjustments were introduced to ensure that the GMA outputs its results in a structured format suitable for training the ANN used in the DEA.

Originally, the GMA only provided a simple mapping between the IDs of re-identified individuals. However, to enable the DEA to learn meaningful patterns, access to both the plaintext PII and their corresponding encodings is required. Therefore, the GMA was extended to output two datasets in the following format:

- For re-identified individuals: `<PII> <encoding> <uid>`
- For non-re-identified individuals: `<encoding> <uid>`

It is important to note that the `uid` is included solely for research and evaluation purposes. It enables researchers to manually track individuals across different processing stages and to assess the performance of the attack. However, in a real-world attack scenario, these `uid`s are neither available nor required. They are entirely excluded from all DEA training and inference steps, ensuring that the attack methodology remains realistic and practically applicable.

In addition to formatting adjustments, certain components of the GMA were removed to streamline the process and reduce unnecessary complexity. Specifically, encoding schemes other than TSH, TMH, and BF were excluded, as the DEA focuses exclusively on these techniques. Other components deemed non-essential, such as graph visualizations and benchmark tests related solely to the GMA, were also removed. This decision was made because the GMA is not the primary focus of this study; its validity and performance have already been established by prior research. These optimizations resulted in a leaner and more efficient attack pipeline, reducing computational overhead while preserving essential functionality.

With these modifications in place, the starting point for the DEA is clearly defined. The attack begins with the two structured datasets. By leveraging this structured output, the DEA can train a machine learning model to probabilistically reconstruct missing n-grams from the encoded records of non-re-identified individuals. The following sections detail the implementation of this approach, including dataset preparation, model architecture, and evaluation strategies.

3.5. Step 2: Data Representation

For an ANN to operate effectively, the input data must be preprocessed into a format compatible with deep learning models. This preprocessing step applies to both the input, encoded representations of PII, and the output, which consists of labels representing the predicted n-grams. Since ANNs in PyTorch operate on tensor-based representations, the transformation of encoded records into tensors is a requirement. This ensures that both re-identified and not-reidentified individuals are structured in a way that enables efficient training and inference.

To facilitate this transformation, custom PyTorch datasets are implemented. These datasets transform the encoded representations into input tensors and represent the associated n-gram labels as binary vectors for multi-label classification, where each position in the label vector indicates the presence or absence of a specific n-gram. This approach enables the model to predict the presence of multiple n-grams per encoded record.

The data representation pipeline is modular and accommodates various encoding schemes, each of which necessitates a tailored preprocessing technique. Depending on the encoding method, such as BF, TSH, or TMH, different strategies are employed to convert the encoded input into tensors while preserving as much information as possible. This ensures that the input is well-suited to the architecture of the corresponding ANN and that the model can effectively learn the mapping between encoded data and plaintext n-grams.

3.5.1. BF Encoding

BF's are fixed-length binary strings, with their length determined by Alice's chosen encoding parameters. The transformation of a BF into a PyTorch tensor is straightforward: each bit in the binary string is directly mapped to a corresponding position in the tensor. This conversion preserves the positions of set bits (i.e., ones), thereby maintaining the structural integrity of the original encoding. The resulting tensor has the same dimensionality as the BF, with ones indicating the activated hash positions and zeros elsewhere. This binary representation serves as the input to the ANN, allowing the model to learn patterns based on the bitwise structure of the encoded PII.

3.5.2. TMH Encoding

TMH, like BFs, produces fixed-length binary bitstrings, with the specific length determined by the encoding parameters selected by Alice. The transformation into a PyTorch tensor mirrors that of the BF: each bit in the TMH string is mapped directly to a corresponding tensor position, preserving the locations of set bits. This direct conversion results in a binary tensor representation that retains the structure of the original TMH encoding. By preserving the positional information of the activated bits, the ANN can effectively learn from the encoded patterns embedded in the TMH representations.

3.5.3. TSH Encoding

The preprocessing of TSH encodings is more complex due to its variable-length representation. Unlike BF and TMH, which produce fixed-length binary bitstrings, TSH generates a set of integers of arbitrary size. This variability arises because columns containing only zero values are dropped during the TSH encoding process.

Since ANNs require fixed-length input vectors, an appropriate transformation is necessary to standardize TSH encodings. Simple aggregation techniques, such as averaging, can lead to substantial information loss, particularly problematic in this already knowledge constrained setting. To preserve the richness of the encoded data, an alternative method is employed to convert TSH encodings into a tensor compatible format.

To achieve this, all unique integer values from both the re-identified and non-reidentified datasets are collected and stored in a set. This set is then sorted in ascending order and transformed into a dictionary that maps each integer to a unique index. Using this mapping, each TSH encoding is converted into a binary vector using a one-hot encoding scheme. For

each integer present in the TSH encoding, the corresponding index in the binary vector is set to one, while all other positions remain zero.

Regardless of the encoding scheme used as input, the output of the ANN remains consistent across all implementations. The model is trained to map the encoded input to a probability distribution over possible n-grams. Thus, the output layer of the ANN performs multi-label classification, predicting the likelihood of each n-gram being present in the original plaintext PII.

3.5.4. Re-Identified Individuals as Labeled Training Data

To enable supervised learning, re-identified individuals are used as labeled training and validation data. Since their PII is known along with their corresponding encoded representation, it is possible to construct datasets where the input consists of transformed encodings (BF, TMH, or TSH, respectively) into tensors and the output labels consist of the correct n-grams derived from the original PII.

To facilitate this process, a predefined dictionary of all possible n-grams is created. This dictionary includes:

- Alphabetical n-grams (e.g., for 2-grams: `aa` to `zz`),
- Numerical n-grams (e.g., for 2-grams: `00` to `99`),
- Alphanumeric mixed n-grams (e.g., for 2-grams: `a0` to `z9`).

Since the datasets used in this research primarily contain first names, last names, and birthdates, these character sets are sufficient to cover the vast majority of n-gram occurrences. Each possible n-gram is mapped to a specific index in the output tensor based on the dictionary, ensuring a consistent label format across all training samples. For example, if index `1` corresponds to the n-gram "ab", and the ANN predicts a 60% probability at index `1`, this is interpreted as a 60% likelihood that "ab" was present in the original plaintext.

By structuring the data in this way, the ANN is trained to learn a mapping from encoded inputs to their corresponding n-gram distributions, enabling the DEA to probabilistically reconstruct plaintext PII from encoded data.

3.6. Step 3: Hyperparameter Optimization

Hyperparameter tuning plays an important role in achieving optimal model performance. Unlike model parameters that are learned during training (e.g., weights and biases), hyperparameters are defined prior to training and control the structure of the model as well as aspects of the learning algorithm. These include architectural choices such as the number of layers as well as training configurations like the learning rate, optimizer, and regularization techniques. Careful selection of these values is important in tasks such as reconstructing plaintext n-grams from encoded representations, where both underfitting and overfitting can lead to substantial performance degradation.

To explore the extensive hyperparameter space efficiently, this work employs Ray Tune, a scalable library for distributed hyperparameter tuning. Specifically, the Optuna search algorithm is used within Ray Tune to guide the optimization process. Optuna leverages a Tree-structured Parzen Estimator, a Bayesian optimization method that prioritizes promising

regions of the search space based on previous trial results. This approach improves search efficiency and reduces the number of iterations required to discover high-performing configurations.

The hyperparameter search space in this study is designed to be both comprehensive and computationally feasible. Key hyperparameters that define the neural network architecture include:

- **Number of hidden layers**: varied between 1 and 4, allowing the exploration of both shallow and deep networks.
- **Hidden layer size**: selected from {128, 256, 512, 1024, 2048}, enabling experiments with compact to large-capacity models.
- **Dropout rate**: sampled uniformly between 0.1 and 0.4 to promote generalization and mitigate overfitting.
- **Activation function**: treated as a categorical variable with options including `ReLU`, `Leaky ReLU`, `GELU`, `ELU`, `SELU`, and `Tanh`. All of these functions introduce non-linearity and aim to mitigate vanishing gradients, but differ in smoothness, output range, and handling of negative inputs.

These architectural parameters create a flexible and expressive search space for discovering well-performing network structures tailored to the task of the DEA.

The optimization strategy is similarly governed by several hyperparameters that influence how the model is trained. The **optimizer** is treated as a categorical hyperparameter, with options including `Adam`, `AdamW`, `RMSprop`, and `SGD`. Each optimizer is paired with a corresponding learning rate sampled from a log-uniform distribution to accommodate the wide sensitivity of models to this parameter. In the specific case of `SGD`, an additional `momentum` parameter is also tuned to control the influence of past gradients in the current weight update.

In conjunction with the optimizer, the choice of a **learning rate scheduler** further enhances the model's ability to converge effectively. The search space for learning rate scheduling strategies includes:

- `StepLR`: reduces the learning rate at fixed epoch intervals,
- `ExponentialLR`: applies exponential decay over time,
- `ReduceLROnPlateau`: reacts to stagnation in validation loss,
- `CosineAnnealingLR`: follows a cosine decay schedule,
- `CyclicLR`: oscillates between lower and upper bounds in modes such as `triangular`, `triangular2`, and `exp_range`.

An additional option to disable learning rate scheduling is also included to assess whether constant learning rates perform better for certain models.

Furthermore, the **loss function** is a hyperparameter, as it directly influences the optimization objective. Several loss functions suitable for multi-label classification are explored:

- `BCEWithLogitsLoss`: a standard binary cross-entropy loss combined with a sigmoid activation, commonly used for multi-label tasks.

- `MultiLabelSoftMarginLoss`: supports probabilistic multi-label targets and is well-suited for scenarios where multiple n-grams may be present simultaneously.
- `SoftMarginLoss`: a generalization of logistic loss for binary classification, which can also be applied to multi-label settings with continuous targets.

This comprehensive optimization configuration enables systematic exploration of training dynamics, ensuring the neural network can effectively learn meaningful mappings for the DEA across various encoding schemes.

Additional parameters are included in the hyperparameter search to fine-tune the model's output behavior and ensure consistent evaluation. A tunable **threshold** parameter, ranging from 0.3 to 0.8, is introduced to convert the model's probabilistic outputs into binary predictions for the presence of specific n-grams. This threshold plays a role in multi-label classification, as it directly affects the balance between precision and recall. Furthermore, the **batch size** used by the data loaders is treated as a tunable parameter, with candidate values of 8, 16, 32, and 64. Varying the batch size allows for a trade-off between computational efficiency and training stability, potentially influencing convergence dynamics and generalization performance.

To ensure a fair and consistent comparison across all hyperparameter configurations, the same training and validation datasets are used in each trial. This controlled setup ensures that variations in performance can be attributed to the model configuration rather than differences in training and validation data.

During tuning, Ray Tune orchestrates multiple parallel trials, each corresponding to a unique combination of hyperparameters sampled from the search space. Optuna's pruning mechanism is also integrated, allowing unpromising trials to be stopped early based on intermediate results (e.g., validation loss), which improves overall efficiency. Performance is evaluated on the validation set, and the best configuration is selected based on a predefined optimization metric, as discussed in Section 3.8.2.

This automated, systematic tuning process ensures that the neural network architecture is well-adapted to the complexity and characteristics of the input encoding. It enables fair comparison across models (BF, TMH, or TSH) and improves both the predictive performance and generalization capability of the reconstruction task.

3.7. Step 4: Model Training and ANN Architecture

The architecture follows a feedforward design and consists of three main components: an input layer, a configurable sequence of hidden layers, and a final output layer. The size of the input layer is determined by the dimensionality of the encoded record. For instance, in the case of the BF and TMH model, the input layer corresponds to the length of the bitstring, which in turn is defined by Alice's chosen parameters. The TSH models define their input layers based on the number of unique integers used across both datasets, the re-identified and non-re-identified individuals.

This modular architecture enables experimentation across different encoding schemes while maintaining a unified framework for training and evaluation.

The hidden layers are structured dynamically, depending on a set of tunable hyperparameters. These include the number of hidden layers and the number of neurons in each layer (hidden layer size). Each hidden layer is followed by a non-linear activation function, enabling

the model to capture complex and non-linear relationships in the data. To mitigate overfitting, dropout is applied after each activation layer, randomly deactivating a fraction of neurons during training. This regularization technique improves the model's ability to generalize to unseen inputs and prevents the memorization of training data.

The output layer remains consistent across all encoding schemes and has a dimensionality equal to the size of the predefined n-gram dictionary. Each output neuron represents the model's predicted probability that a specific n-gram appears in the original plaintext record. Given that multiple n-grams may be present in a single encoded record, the task is framed as a multi-label classification problem. Therefore, the output layer uses a sigmoid activation function, allowing the network to assign independent probability estimates to each n-gram.

This modular architecture is implemented using PyTorch's `nn.Sequential` API, which enables a clean, maintainable, and extensible model definition. Furthermore, this design supports efficient hyperparameter optimization, as components, such as the number of layers, hidden layer size, activation function, and dropout rate, can be systematically varied across experimental runs. By exploring this hyperparameter space, the model can be tailored to maximize reconstruction performance for each specific encoding scheme.

3.7.1. Foundations of Neural Network Success in DEA

Attempting to reconstruct plaintext information from encoded representations based on hash functions presents a challenge due to the nature of cryptographic hashing. Since hash functions are designed as one-way functions, reversing the transformation to recover the original input is theoretically infeasible. However, while exact reconstruction is not possible, a probabilistic approach can still be employed to infer likely plaintext components based on statistical patterns within the encoded data.

ANNs provide a framework for learning complex mappings between input encodings and output predictions, making them well-suited for this task. The function of the ANN in the context of the DEA is to predict n-grams by learning from re-identified individuals, those whose plaintext information is known alongside their corresponding encodings. Through this supervised learning process, the model captures frequency patterns that emerge due to the deterministic nature of the encoding process. In essence, the ANN learns which n-grams are statistically associated with specific positions or patterns in the encoded representations and leverages these associations to estimate their likelihood in unseen encoded inputs.

Although hash functions introduce collisions, where different inputs may produce the same hash output, the ANN can still extract meaningful probabilistic insights by generalizing over these mappings across many samples. This enables the DEA to output a ranked list of likely n-grams per record, thereby forming the basis for the reconstruction of PII from encoded data in a probabilistic, frequency-informed manner.

This is possible due to the fact that for names certain n-grams are more likely to occur than others. For example, n-grams like "an", "el", or "ar" are common in many names, while others like "xz" or "qv" are rare. By learning these statistical patterns, the ANN can prioritize more probable n-grams during reconstruction, improving the overall accuracy of the attack.

3.7.2. Training the Model

To effectively train and evaluate the ANN model with the best hyperparameters, the dataset is divided into three distinct subsets: a training set, a validation set, and a test set. The

training set consists of 80% of the labeled dataset, while the remaining 20% is designated as the validation set. The test set comprises the not-reidentified individuals, serving as the primary evaluation set for the trained model.

Dataloaders are created for each of these subsets to facilitate efficient mini-batch processing. Different batch sizes are employed depending on the chosen hyperparameter value to optimize computational performance and convergence behavior. The training and validation dataloaders enable efficient iteration over the respective data splits, ensuring that the ANN is exposed to all available samples during training and validation.

The training process consists of multiple epochs, where each epoch involves iterating through the entire training dataset using the data loader. For each mini-batch, the model performs a forward pass, computes the loss, and applies backpropagation to update the network's parameters using the selected optimizer. If a learning rate scheduler is specified, it is applied according to its strategy to dynamically adjust the learning rate throughout training.

After processing all training batches in an epoch, the model's performance is evaluated on the validation set by computing the validation loss and selected performance metrics. This allows the training loop to monitor potential overfitting and adjust training accordingly, for instance by applying early stopping or learning rate decay. Throughout this process, the model learns the optimal weights and biases that minimize the loss on the training set, guiding it toward better generalization and performance.

3.8. Step 5: Application to Encoded Data

3.8.1. Performance Evaluation

Once the ANN is trained and validated, it can be applied to the non-re-identified individuals, whose encoded representations were not matched during the GMA step. The performance of the DEA is evaluated on this test set. Several metrics are computed to assess the effectiveness of the attack, including precision, recall, F1-score, and the Dice similarity coefficient.

Precision quantifies the proportion of correctly predicted n-grams among all n-grams predicted by the model. It reflects the model's ability to avoid false positives during the reconstruction of plaintext features from encoded representations. A high precision value indicates that most of the predicted n-grams are indeed part of the original PII, suggesting a low rate of over-generation. This is particularly relevant in the context of DEAs, where the specificity of reconstructed values is to minimize erroneous inferences. Formally, for a given record, let T be the set of true n-grams and P the set of predicted n-grams. The precision is defined as:

$$\text{Precision} = \frac{|T \cap P|}{|P|} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

where TP denotes the number of true positives and FP the number of false positives. If $|P| = 0$, precision is defined as 0 to avoid division by zero.

Recall measures the proportion of true n-grams that were successfully predicted by the model. It captures the completeness of the reconstruction, indicating how much of the original information was extracted from the encoded data. A high recall implies that the attack is capable of recovering a fraction of the original content, even if some incorrect n-grams are also included. The recall is defined as:

$$\text{Recall} = \frac{|T \cap P|}{|T|} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

where FN represents the number of false negatives (i.e., true n-grams missed by the model). If $|T| = 0$, recall is defined as 0.

F1-score is the harmonic mean of precision and recall, providing a single score that balances both correctness and completeness. It is informative in scenarios where the number of predicted and ground-truth n-grams may differ substantially, preventing either precision or recall from dominating the performance evaluation. The F1-score is defined as:

$$\text{F1-score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

If both precision and recall are zero, the F1-score is defined as zero.

The **Dice similarity coefficient**, which is mathematically equivalent to the F1-score for binary sets, is computed as twice the size of the intersection of the predicted and actual n-gram sets divided by the total number of elements in both sets. In the DEA setting, the Dice coefficient serves as an interpretable and robust measure of set overlap. It is well-suited for evaluating partial reconstructions, where perfect recovery may be infeasible, but alignment with the ground truth still reflects successful inference.

Each of these metrics is computed on a *per-record* basis, comparing the predicted and true n-gram sets for every individual sample. To evaluate the model's performance on the entire dataset, the scores are then averaged across all records. This approach ensures a granular and interpretable assessment of the reconstruction quality, accounting for varying degrees of difficulty across different entities.

3.8.2. Choosing the Right Metric for Hyperparameter Optimization

The performance of the neural network employed in the DEA is evaluated using several metrics: precision, recall, F1-score, and the Dice similarity coefficient. While each metric provides valuable insight into the quality of plaintext reconstruction, only one can be selected as the optimization target during hyperparameter tuning (e.g., via Ray Tune). From the attacker's perspective, this choice is strategic, as it directly influences the re-identification behavior and the nature of the reconstructed information.

Optimizing for Precision: Conservative but High-Confidence Inference Optimizing for precision encourages the model to prioritize correctness over completeness. In this configuration, the model is incentivized to predict fewer n-grams but with high certainty that each predicted token is truly part of the original plaintext. This reduces the risk of false positives, which is important when re-identified individuals inform downstream actions, such as attempting identity theft or inferring sensitive attributes. A precision-oriented attacker thus obtains fewer, but more reliable, re-identifications, minimizing noise in the reconstructed dataset. However, this strategy may overlook harder-to-recover but valid features.

Optimizing for Recall: Aggressive and Broad Coverage In contrast, recall optimization emphasizes completeness, seeking to recover as many original n-grams as possible, even at the cost of accuracy. This strategy is advantageous when the attacker aims to maximize information

gain, such as in exploratory analysis or probabilistic linkage. High recall ensures broader coverage of the encoded dataset and may benefit post-processing or correction steps filtering out wrongly predicted n-grams. However, it also increases the risk of false positives, potentially degrading the overall quality of re-identifications.

Optimizing for F1-Score or Dice: Balanced Inference The F1-score and the Dice similarity coefficient (which are mathematically equivalent for binary sets) provide a harmonic balance between precision and recall. Optimizing for these metrics helps prevent predictions that are either too conservative or too permissive. Since the attacker typically cannot externally validate predictions, balancing precision and recall mitigates the risk of errors. Additionally, the Dice coefficient provides an intuitive and interpretable measure of similarity,

Strategic Consideration Ultimately, the attacker's choice of optimization metric should reflect the intended downstream use of the reconstructed data. If re-identified tokens are used to trigger sensitive actions, such as identity theft or inferring information about an person, then precision is the preferred metric, as it minimizes the risk of false positives. In contrast, when the goal is to maximize overall data leakage or to support probabilistic analyses, recall becomes more advantageous, as it emphasizes completeness over certainty. For scenarios that require a balanced evaluation of both precision and recall, particularly when no strong bias toward one type of error exists, metrics such as the F1-score or Dice coefficient provide a robust and interpretable compromise.

In this study, the Dice similarity coefficient was selected as the primary optimization objective during hyperparameter tuning, due to its balanced nature and its interpretability in the context of set overlap. This reflects the attacker's goal of performing broadly effective and consistent reconstructions, maximizing utility while minimizing both false positives and false negatives.

3.8.3. Results

The result of the ANN inference is a set of predicted n-grams for each encoded record in the test set. These predictions are generated based on the learned patterns from the training data, where the model has been exposed to both the encoded representations and their corresponding plaintext n-grams. As a probability for each n-gram is output, a threshold is applied to convert the predicted probabilities into binary predictions. This thresholding process determines whether a specific n-gram is considered present in the original plaintext or not. The choice of threshold is important, as it directly influences the precision and recall of the model's predictions. A higher threshold may yield more conservative predictions, resulting in higher precision but lower recall, while a lower threshold may increase recall at the cost of precision. The threshold is typically set based on the desired balance between precision and recall, depending on the attacker's objectives.

The result is therefore a set of predicted n-grams for each encoded record, which can be further processed or analyzed to extract meaningful information. The predicted n-grams can be used to reconstruct the original plaintext PII by combining the predicted tokens into coherent strings.

3.9. Step 6: Refinement and Reconstruction

The subsequent step focuses on reconstructing interpretable plaintext attributes, such as first name, surname, and date of birth, from the sets of overlapping n-grams predicted by the ANN following a threshold-based filtering process. This reconstruction is essential for empirically demonstrating the extent to which encoded identifiers can be reversed into human-readable information, thereby highlighting the problem of privacy vulnerabilities in PPRL systems.

To tackle this task, three distinct reconstruction strategies are explored, each offering different trade-offs in terms of computational complexity, accuracy, interpretability, and the level of certainty regarding the correctness of the reconstructed identifiers. These strategies are designed to simulate varying levels of attacker sophistication, from basic structural heuristics to dictionary guided and machine learning assisted inference, thereby providing a comprehensive perspective on the practical threat posed by DEA.

The first method, a graph-based approach, serves as a structural baseline and operates in a fully deterministic manner. It constructs a directed graph in which each node represents a character, and each edge corresponds to a predicted n-gram that links the starting character to the ending character. The reconstruction task is framed as finding the longest paths through this graph, under the assumption that sequences having the greatest number of predicted n-grams are most likely to represent complete and meaningful strings. These paths are then interpreted as candidate identifiers.

This approach is computationally efficient and independent of external resources, but it does not resolve ambiguities when multiple equally long or plausible paths exist. Moreover, it lacks semantic validation mechanisms to verify whether the reconstructed substrings resemble real-world values, which limits its practical effectiveness in realistic attack scenarios. On the other hand, it offers the highest certainty regarding reconstruction correctness among the evaluated methods, as it relies solely on the predicted n-grams without introducing any additional inference or external assumptions. This makes it a transparent and interpretable baseline, particularly suitable for isolating the contribution of the n-gram prediction model itself.

The second approach incorporates prior knowledge through a dictionary-based fuzzy matching technique. A curated reference list of known names, comprising frequently occurring first names and surnames, is preprocessed into corresponding sets of n-grams. These name lists are sourced from the U.S. Census and the Social Security Administration, providing realistic and demographically representative coverage of common naming patterns. For each predicted set of n-grams, candidate names from the dictionary are evaluated using similarity metrics such as the Dice or Jaccard coefficient. The top-scoring candidates are selected as reconstruction hypotheses, prioritizing entries with the greatest n-gram overlap.

This method offers increased interpretability by anchoring the reconstruction process in semantically valid, real-world values and provides a degree of robustness against structural noise or incomplete n-gram sets. However, its effectiveness is inherently limited by the quality and coverage of the dictionary. It may fail to reconstruct rare or out-of-vocabulary names and implicitly assumes that personal names adhere to those found in the predefined list.

The third and most flexible strategy employs a generative language model to reconstruct attributes directly from the set of predicted n-grams. In this approach, the unordered collection of n-grams is embedded into a natural language prompt, allowing the model to infer likely combinations of names and birthdates. Leveraging its ability to synthesize coherent text and recognize semantic relationships, the language model can resolve ambiguities, infer structural patterns, and even compensate for missing or noisy n-grams.

While this method offers the greatest expressive power and adaptability, it also introduces notable limitations in terms of reproducibility, transparency, and correctness. The model may produce outputs that are fluent but factually incorrect (i.e., hallucinations), and there is no guarantee that all input n-grams are represented in the reconstructed response. As such, this approach is best employed as a heuristic aid to guide human interpretation, rather than as a deterministic component of the reconstruction pipeline.

Together, these three reconstruction strategies offer complementary perspectives: the first illustrates the structural feasibility of assembling plausible tokens from n-gram fragments, the second emphasizes semantic validation through prior knowledge, and the third showcases the generative capabilities of modern language models. A comparative evaluation of their outputs enables a nuanced assessment of the threats posed by DEAs, illustrating how attackers with varying levels of knowledge, computational resources, and methodological assumptions could effectively compromise encoded personal data.

3.9.1. Directed Graph Based Reconstruction

The first reconstruction strategy is based on modeling the set of predicted n-grams as a directed graph to recover plaintext attributes. In this representation, each character becomes a node, and each n-gram is modeled as a directed edge from its first character to its last. The resulting graph is typically simple, though it can be either acyclic or cyclic depending on the structure and overlap of the predicted n-grams. Reconstruction is then formulated as a longest path problem within this graph. The central assumption is that the longest possible path, in terms of included n-grams, is most likely to represent the most complete and coherent reconstruction of the original plaintext string.

In the case of a Directed Acyclic Graph (DAG), the longest path can be computed efficiently using well established graph traversal algorithms. Specifically, this implementation leverages NetworkX's built-in `dag_longest_path` function, which operates in linear time with respect to the number of nodes and edges ($\mathcal{O}(V+E)$), making it suitable for small to moderately sized graphs typical in this setting.

When the graph contains cycles, however, the problem becomes more complex, as naive traversal may result in infinite loops. To handle this, a custom recursive Depth First Search (DFS) Search strategy is employed. The algorithm explores all reachable paths from each starting point while maintaining a set of visited edges to avoid revisiting the same n-gram multiple times within a single path. At each step, the current reconstruction string is extended with the next character, and the longest sequence encountered during traversal is stored.

An illustrative example is shown in Figure 3.2, based on the set of 2-grams `["jo", "oh", "hn", "do", "oe", "be", "ha", "af", "st"]`, which forms a directed graph with eleven nodes. Since the graph is cyclic, the algorithm performs a recursive DFS, tracking visited edges to prevent infinite loops. It initiates the search from each node, extending candidate paths by appending the next character at each step. The longest path found in this process corresponds to the string "johndoe", as all alternative paths result in shorter sequences.

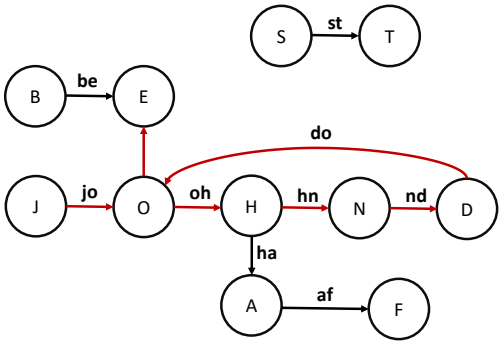

Figure 3.2.: Example reconstruction using the Directed Graph based approach.

This method is fully deterministic and does not rely on any external data sources, making it flexible and interpretable. However, it still faces challenges in resolving ambiguities when multiple equally long paths exist and lacks any semantic validation to ensure that the output resembles real-world names or values. Nonetheless, it provides a strong structural baseline that emphasizes maximal use of the predicted n-gram set.

From a computational perspective, this approach is efficient for small graphs, as typically encountered in n-gram-based reconstructions of individual fields such as names. For DAGs, the runtime remains linear, and even in the presence of cycles, the DFS based implementation remains tractable due to the limited number of characters and n-grams per input record. In practice, most graphs contain fewer than 30 edges, and recursion depth is shallow. The approach scales well to batch processing across many records and offers a favorable trade-off between accuracy and runtime, making it a practical first stage reconstruction strategy.

3.9.2. Dictionary Based Reconstruction

The dictionary based reconstruction strategy leverages curated datasets of common attributes like first names, surnames, and birthdates to infer plaintext attributes from predicted n-grams. For each predicted record, the method computes a similarity score between the predicted set of n-grams and the n-gram representations of each dictionary entry. This is performed separately for each attribute, e.g. first name, surname, and birthdate, by comparing the predicted n-grams to those extracted from the corresponding dictionaries. The top scoring candidates are selected as reconstruction hypotheses, prioritizing entries with the greatest n-gram overlap as measured by a similarity metric, such as the Dice coefficient.

This approach increases interpretability by grounding predictions in semantically valid and human readable values. It is robust against noise or partial n-gram sets, as it can still return plausible results even when only a subset of the correct n-grams is present. Moreover, by using separate dictionaries for each attribute, the method is capable of multi-field reconstruction,

making it more versatile and scalable than methods that only target a single attribute.

Reconstruction is conducted sequentially across multiple attributes. First, the predicted n-grams are matched e.g. against a list of known first names. Once the most likely match is found, its n-grams are removed from the candidate set to avoid double counting when reconstructing the next attribute. This is particularly useful in cases where certain attributes like names can appear both as first and last names (e.g., "James"). A similar strategy is then applied to the remaining attributes.

The quality of this reconstruction method is highly dependent on the coverage and granularity of the dictionaries used. Larger and more representative dictionaries improve the likelihood of accurate matches, while rare or culturally diverse names may remain underrepresented. Additionally, the similarity scoring mechanism plays a key role. The Dice coefficient is used due to its effectiveness in measuring set overlap between short strings, such as n-gram sets.

The runtime complexity of the dictionary-based reconstruction strategy is primarily determined by the number of predicted entries and the size of the reference dictionaries.

For each of the n predicted entries, the algorithm performs similarity comparisons against all candidates in three separate dictionaries: given names, surnames, and birthdates. Let D_{given}, D_{surname}, and D_{birthday} denote the number of entries in each respective dictionary. The total number of comparisons per entry is then

$$D = D_{\text{given}} + D_{\text{surname}} + D_{\text{birthday}}.$$

The similarity metric used for comparison is the Dice coefficient over n-grams, which can be computed in linear time with respect to the number of n-grams in the compared tokens. Assuming an average of g n-grams per token, the runtime complexity of computing a single Dice similarity is $\mathcal{O}(g) = \mathcal{O}(|A| + |B|)$ where $|A|$ and $|B|$ are the number of n-grams in the two sets being compared.

Therefore, the overall runtime complexity of the dictionary-based reconstruction attack is

$$\mathcal{O}(n \cdot D \cdot g)$$

.

In practice, the values of g are relatively small, as the average number of 2-grams per name is limited (typically between 4 and 10). Therefore, the reconstruction remains efficient even for large n. Furthermore, this approach is trivially parallelizable across entries, making it well suited for batch processing in realistic attack scenarios.

Overall, this strategy provides a realistic and reproducible method for attackers to reconstruct personal information, particularly in scenarios where the attacker has access to auxiliary data such as population wide name lists or public birthdate datasets. By exploiting structural patterns and common token distributions, this method demonstrates how dictionary guided attacks can ly enhance the re-identification capabilities.

3.9.3. Generative Language Model Based Reconstruction

The final and most flexible reconstruction strategy explored in this thesis involves leveraging Large Language Model (LLM)s to infer original identifiers from predicted n-grams. Unlike previous approaches, which operate within constrained matching logic, LLMs can reason over partial, noisy, or ambiguous input and generate semantically coherent completions based on prior knowledge. In this context, the model is prompted with a batch of predicted n-gram

sets and asked to reconstruct corresponding attribute values such as given names, surnames, or birthdates.

This approach proves particularly robust in cases where the n-gram predictions are incomplete or include noise. Owing to their generative and contextual capabilities, LLMs can infer plausible attribute values even when important information is missing. Additionally, the model can implicitly correct for common errors or insert culturally plausible completions, making it attractive for reconstructing personal identifiers under uncertainty. This flexibility, however, comes at the cost of reproducibility and transparency. Since LLMs are non-deterministic, their responses may vary across repeated executions, even with identical input. Furthermore, hallucinations, plausible but incorrect outputs, can occur when the model overgeneralizes or encounters ambiguous prompts.

Another practical limitation concerns model availability and cost. If the attacker does not have access to a self-hosted LLM, they must rely on external APIs (e.g., OpenAI or similar providers), which introduces cost and latency. Moreover, results can vary substantially depending on the specific model used (e.g., GPT-3.5, GPT-4, open-source variants) and on the design of the prompt. To address the efficiency of batch processing, this thesis adopts a strategy in which 15 n-gram sets are reconstructed in parallel per prompt call, a trade-off found to balance cost, latency, and response quality effectively.

The prompt used for this reconstruction task is designed to guide the model toward structured attribute extraction while accommodating the free-form nature of language generation. The prompt template can be found below and should be considered an integral part of the reconstruction method:

Prompt Template for LLM Based Reconstruction

You are an attacker attempting to reconstruct the **given name**, **surname**, and **date of birth** of multiple individuals based on predicted 2-grams obtained through a dataset extension attack.
Each individual is represented by a `uid` and a list of 2-grams. For each entry, infer:

- `<GIVEN_NAME>`
- `<SURNAME>`
- `<BIRTHDATE>` (in `M/D/YYYY` format, without leading zeros)

Only return **valid JSON** in the following format:

```
[
  {
    "uid": "29995",
    "<GIVEN_NAME>": "Leslie",
    "<SURNAME>": "Smith",
    "<BIRTHDATE>": "12/22/1974"
  },
  ...
]
```

Here is the input:

```
{
  "29995": ["Le", "es", "sl", "li", "ie", ...],
  ...
}
```

After submitting the prompt, the model typically responds within several seconds, depending on the provider and load conditions. The output is then post-processed to extract the structured attribute values and evaluated using standard string similarity metrics. While this approach is not used in the final evaluation due to its lack of reproducibility and its dependency on external infrastructure, it represents a direction for future attack strategies, especially in human-in-the-loop or investigative contexts where creative reconstruction is desirable.

4. Results

4.1. Experiments

To evaluate the effectiveness of the previously defined DEA, a series of experiments are conducted using multiple datasets. These experiments aim to assess the attack's performance across different encoding schemes and datasets using different execution settings to analyze its ability to reconstruct plaintext information from encoded identifiers.

The primary dataset used is the `fakename` dataset, which is synthetically generated using the American name set provided by the Fake Name Generator. This dataset was previously employed in related work by Schaefer et al. [SAH24], making it a suitable benchmark for comparative evaluation. It includes realistic combinations of personal identifiers and is well-suited for testing the scalability and reliability of both the GMA and DEA pipelines.

The `fakename` datasets consist of synthetically generated entries, each containing a given name, surname, and date of birth. These datasets aim to resemble realistic combinations of personal identifiers while ensuring privacy and reproducibility. For evaluation purposes, multiple dataset instances of varying sizes are used: 1,000, 2,000, 5,000, 10,000, 20,000, and 50,000 entries.

The primary advantage of using this dataset family lies in its scalability. By maintaining a consistent schema while varying the number of records, the impact of dataset size on the performance and success of the DEA can be systematically analyzed. This enables controlled experiments that highlight how the quantity of available data influences re-identification, training quality, and generalization performance of the attack models.

An additional dataset used in this study is the `euro_person` dataset provided as part of the simulated data for the ESSnet DI on-the-job training course on record linkage, held in Southampton from 25–28 January 2011. The dataset was created by Paula McLeod, Dick Heasman, and Ian Forbes from the UK Office for National Statistics and contains realistic, fictionalized personal information intended for the training and evaluation of record linkage techniques. The `euro_person` dataset includes forename (`PERNAME1`), surname (`PERNAME2`), and full date of birth composed of day, month, and year, which were concatenated into a single `DOB_FULL` attribute for the purposes of this work. The dataset consist of 26.625 records. As the dataset also serves as a ground-truth reference for other simulated sources such as Census, CIS, and PRD, it is well-suited for evaluating the precision and completeness of plaintext reconstruction and re-identification in the context of the DEA.

In addition to the synthetic and benchmark datasets, this thesis also incorporates a curated version of the Titanic passenger manifest, referred to as `titanic_full`. This dataset consists of 891 unique records and includes the fields `firstname` and `surname` used for internal tracking. While not originally intended for record linkage evaluation, the dataset offers a semi-realistic collection of personal identifiers derived from historical records. It provides a useful test case for examining the impact of natural name diversity, varying name lengths, and non-standard naming formats (e.g., inclusion of titles or parenthetical information) on the performance of both Graph Matching and Dataset Extension Attacks. Due to the historical and English centric

nature of the data, it shares some limitations with other Western-focused datasets used in this work but nonetheless adds valuable variety in terms of name structure and frequency.

The experiments are conducted across a range of settings and scenarios to comprehensively evaluate the effectiveness of the DEA. Each encoding scheme, namely BF, TSH, and TMH, is tested individually across all datasets as described earlier. This allows for a direct comparison of reconstruction performance under different privacy-preserving encoding mechanisms.

To additionally analyze how the quantity of the training data affects the DEA, the preceding GMA step is executed with varying levels of overlap between Alice's and Eve's datasets. For each dataset and encoding scheme, the GMA is run multiple times with overlap ratios ranging from 20% to 80%, in increments of 20%. This simulates different real world scenarios where the attacker has access to varying amounts of auxiliary information. The resulting re-identifications from the GMA then serve as the labeled training data for the DEA, thus allowing for a detailed evaluation of how overlap levels influence overall reconstruction success.

In addition to varying dataset sizes and overlap levels, different attacker scenarios are considered by employing different drop from strategies to evaluate the robustness of the DEA under more and less realistic assumptions. The first scenario, Eve's auxiliary dataset D_e is a strict subset of Alice's dataset D_p, i.e., $D_e \subseteq D_p$. In this case, the overlap o is defined as the ratio $o = \frac{|D_e|}{|D_p|}$. The elements in D_e are generated by randomly sampling $|D_e| = \lfloor o \cdot |D_p| \rfloor$ records from D_p without replacement. While this setup simplifies evaluation and isolates the impact of training data availability, it is also highly idealized and does not reflect the complexity of real world linkage scenarios.

To address this, a second, more realistic setting is also considered, where both D_p and D_e contain disjoint as well as overlapping individuals. That is, $D_e \nsubseteq D_p$, but $D_e \cap D_p \neq \emptyset$. In this scenario, the auxiliary and target datasets each include individuals not present in the other, simulating cases where Eve has partial but non exclusive knowledge of the data. This setup introduces additional challenges for both the GMA and DEA, as structural mismatches and auxiliary noise may degrade re-identification and reconstruction accuracy.

This setup mirrors the experimental methodology employed by [SAH24], ensuring consistency and comparability with prior work on the GMA. By varying the overlap rate and dataset composition in this way, a diverse range of re-identification scenarios is created, which directly impacts the amount and quality of training data available for the DEA. This, in turn, enables a systematic evaluation of the DEA's ability to generalize from partially re-identified data. As the GMA identifies different subsets of individuals under varying overlap conditions, the resulting re-identification sets are used to train the neural network, while the remaining non matched records serve as the test set. Thus, each experiment yields a distinct train-test split, providing a rich basis for assessing the reconstruction capabilities of the DEA under different supervision levels and graph-matching outcomes.

For the DEA specific configuration, several fixed settings were employed to ensure comparability across all experimental conditions. First, the dataset of re-identified individuals, used as labeled training data, was split into training and validation sets using a fixed 80/20 ratio. This choice reflects common machine learning practice and provides a balanced compromise between model learning and validation reliability.

One of the most critical components of the DEA pipeline is the hyperparameter optimization step, which is responsible for identifying the most effective neural network architecture. For this purpose, a total of 125 trials were conducted for each experimental setting. This number was chosen to provide sufficient coverage of the hyperparameter space while maintaining

computational feasibility.

Each trial, as well as the final training run for the best performing model, was limited to a maximum of 20 training epochs. While this represents a relatively high upper bound, overfitting is mitigated through the use of early stopping. Specifically, training was halted if the validation loss did not improve for five consecutive epochs (patience = 5), with a minimum delta of $1 \cdot 10^{-4}$ required to qualify as an improvement. This strategy ensures both efficient training and effective model selection, especially when performance plateaus early.

The search space for the hyperparameter optimization follows the configuration described in Section 3.6. Throughout the entire DEA pipeline, the **Dice coefficient** is used as the objective metric for optimization. This choice is motivated by its robustness and balanced nature, as it integrates both precision and recall and has consistently yielded the most promising results during preliminary manual testing.

For efficient optimization, the hyperparameter search is executed using $n - 1$ CPU cores, where n is the number of available logical processors. Additionally and optionally an NVIDIA GPU can be used to accelerate the training of the neural networks during hyperparameter optimization. This allows for near maximal parallelism during hyperparameter tuning, reducing the total runtime without compromising system stability.

In the final re-identification phase, two reconstruction strategies are evaluated to enable comparative analysis: (1) the greedy, graph-based reconstruction method described in Section 3.9.1, and (2) the dictionary-based fuzzy matching approach described in Section 3.9.2. Both methods are deterministic and computationally efficient, making them suitable for large scale experimental evaluation.

The language model based reconstruction method is deliberately excluded from the evaluation. Despite showing potential in early qualitative testing, its dependence on proprietary models, token-based pricing, and limited reproducibility make it unsuitable for scalable and reproducible experimentation within the current research setting.

All experiments were conducted on a virtual machine running Ubuntu 24.04, equipped with 20 cores of a virtualized AMD processor (QEMU Virtual CPU version 2.5+). The system was provisioned with 176 GB of RAM and featured an NVIDIA GeForce RTX 3090 Ti GPU with 24 GB of dedicated VRAM.

This high-performance computing setup enabled efficient parallel execution of the hyperparameter optimization trials and accelerated training of the neural networks via GPU. The extensive memory capacity was particularly beneficial during dataset preprocessing and batch-wise loading of large datasets, ensuring that all encoding schemes and reconstruction strategies could be evaluated without resource bottlenecks.

4.2. Evaluation Metrics

The performance of the DEA is assessed using several metrics that are systematically recorded throughout the experimental runs.

Although the DEA constitutes an offline attack, meaning the adversary can operate without time constraints once both encoded datasets are available, the runtime of the attack remains a valuable indicator of its practical feasibility. Therefore, the total runtime as well as the runtime of each individual stage within the DEA pipeline (e.g., data preprocessing, model training, inference, and reconstruction) is measured and documented.

A central metric for assessing the effectiveness of the attack is the *re-identification rate*. This

metric is defined as the number of individuals that are successfully and correctly re-identified by the DEA, divided by the total number of individuals who were not matched during the initial GMA. A successful re-identification in this context means that the DEA is able to fully reconstruct the original plaintext attributes of a record such that it exactly matches a record in Alice's encoded dataset. Thus, the re-identification rate reflects the proportion of previously unmapped individuals that the attacker can recover using the inference based approach.

Another important aspect of evaluating the DEA is its performance in predicting the correct n-grams, which form the basis for reconstructing plaintext attributes. To assess the quality of these predictions, several standard classification metrics are con, namely, precision, recall, and the F1-score.

Precision measures the proportion of correctly predicted n-grams among all predicted n-grams, thereby quantifying the model's ability to avoid false positives. Recall, on the other hand, captures the proportion of true n-grams that were successfully recovered, reflecting the completeness of the reconstruction. The F1-score, which is the harmonic mean of precision and recall, offers a balanced assessment of the model's correctness and coverage.

In the context of this work, the F1-score is of particular interest, as it has shown to be the most reliable metric for quantifying n-gram-level performance. Moreover, it is mathematically equivalent to the Dice similarity coefficient for binary sets, making it not only interpretable but also consistent with the optimization objective used during the hyperparameter tuning stage (cf. Section 4.1). This alignment ensures that the evaluation metric reflects the actual optimization goal of the DEA.

To enable a meaningful comparison, the performance of different DEA configurations is evaluated not only against each other but also against a baseline strategy. This baseline serves as a lower bound for the expected prediction quality and helps contextualize the improvements achieved through the proposed DEA attack pipeline.

The baseline approach simulates an attacker who, for each non re-identified individual, simply predicts the k most frequent n-grams across the entire dataset. The value k is equal to the average length of an entry minus one, as the n-grams are overlapping. This assumes that the attacker has full access to the distribution of n-grams in the encoded dataset, a reasonable assumption in a research setting, where the dataset is known, but less realistic in real-world attacks. Still, it provides a practical lower-bound for evaluation.

An analysis of the `fakename` datasets revealed that the average total length of a full entry, comprising first name, surname, and date of birth, is approximately 21 characters. Given that 2-grams are used for encoding, this corresponds to roughly 20 overlapping 2-grams per entry. Consequently, the baseline is defined by selecting the top $k = 20$ most frequent 2-grams across the entire dataset and predicting this fixed set for every test record. This naive method disregards any record specific characteristics and instead reflects a population wide frequency-based guess, serving as a simple yet informative lower bound for comparison.

The result is a dataset specific set of baseline metrics. These are mainly precision, recall, F1-score, and Dice similarity, against which the DEA model's predictions can be compared. These baseline metrics vary with dataset size, since the n-gram frequency distribution shifts with the number of records. The aggregated results of the baseline performance across datasets are visualized in Figure 4.1.

As illustrated in Figure 4.1, the guessing-based baseline yields relatively stable performance across increasing dataset sizes. While precision values remain low, around 0.215, recall consistently exceeds 0.245, reflecting that common n-grams are disproportionately favored by the baseline strategy. This results in F1 scores clustering near 0.230, with similarly low Dice sim-

ilarity scores. Notably, the limited variance across dataset sizes suggests that the baseline's effectiveness is only marginally impacted by the scale of the dataset, despite shifts in n-gram frequency distributions. These findings reinforce the role of the baseline as a simple, size-agnostic lower bound against which more sophisticated, learning-based DEA models can be benchmarked.

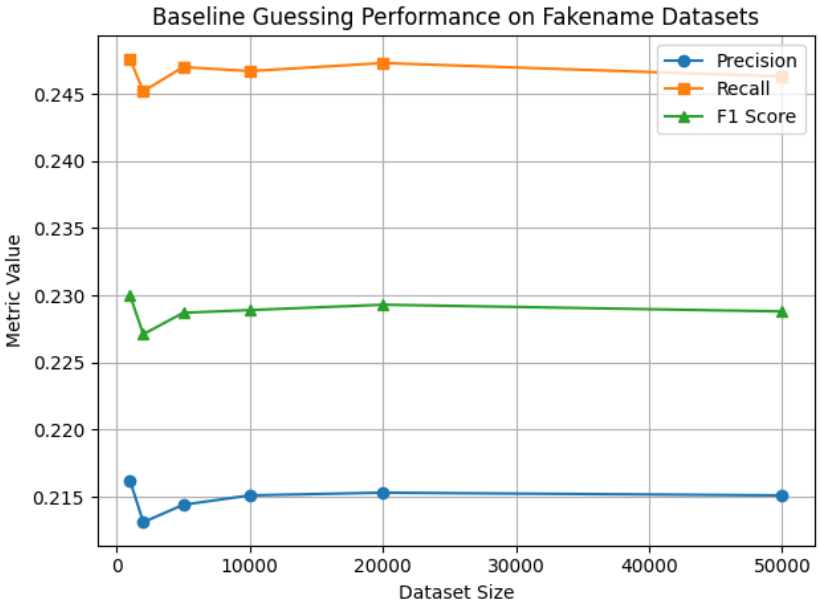

Figure 4.1.: Evaluation of the baseline performance on the `fakename` dataset: For each dataset size, the prediction quality of the 20 most frequent 2-grams is shown in terms of **precision**, **recall**, and **F1-score**. The average entry length is 21 characters.

For the `euro_person` dataset, the baseline strategy for evaluating the effectiveness of the DEA follows the same procedure as for the `fakename` dataset. Specifically, the k most frequent n-grams in the training data are used to simulate a naïve guessing approach. Based on the analysis of this dataset, the average length of a full entry, comprising forename, surname, and full date of birth, is approximately 20 characters. Assuming the use of 2-grams with overlapping character windows, this corresponds to around 19 distinct 2-grams per entry. As a baseline, the top 19 most frequent 2-grams are selected and uniformly predicted for each record, independent of the individual characteristics of the entries.

The resulting performance metrics for this baseline prediction indicate a precision of 0.2197, a recall of 0.2446, and an F1-score of 0.2306. These values provide an important reference point for evaluating the added value of the DEA pipeline, as they quantify the effectiveness of a purely frequency driven reconstruction approach.

For the `titanic_full` dataset, the average length of a full entry (consisting of given name and surname) is approximately 26 characters, indicating relatively high complexity due to longer and often multi-token names. Therefore the top 25 most frequent 2-grams are selected as the baseline for reconstruction. The baseline reconstruction approach achieved a precision of 0.2468, a recall of 0.3770, and an F1 score of 0.2896. These modest performance values reflect the structural challenges posed by the dataset, including the presence of honorifics, compound names, and non-standard formatting. The results suggest that even a naïve guessing strategy

can partially recover plaintext identifiers, but also establish a meaningful lower bound for evaluating the performance improvements of learning-based approaches such as the DEA.

Notably, for the `titanic_full` dataset, the overlap ratio used in the experiments was adjusted to the set {0.2, 0.4, 0.6, 0.7, 0.8, 0.9}. This adaptation was necessary because the dataset is relatively small, and the GMA fails to identify any individuals at lower overlap values. Meaningful results only begin to emerge at an overlap of 0.7 or higher.

Notably, the similarity of the baseline metrics to those obtained on the `fakename` and `euro_person` dataset highlights the generality of the method across synthetic and semi-realistic datasets, further justifying its inclusion as a comparative benchmark.

4.3. Analysis

In this section the results of the DEA experiments are presented and analyzed. The analysis is structured into several subsections, each focusing on a specific aspect of the DEA performance. As described in Section 4.1, the experiments were conducted across multiple datasets, encoding schemes, overlaps and drop from strategies. Therefore, several of these configuration settings will be fixed to analyze the impact of the remaining parameters on the DEA performance.

One important aspect especially for smaller datasets and overlap sizes is, that it could occur that the GMA does not identify any individuals, i.e., the re-identification set is empty. In this case, the DEA is not able to reconstruct any plaintext information. This is the case because if there are no re-identified individuals in place, there is no training data for the DEA available. Therefore the results of these experiments are not reported and the corresponding data points are excluded from the analysis and plots.

4.3.1. TMH

The following subsection focuses on the results obtained using the TMH encoding scheme. In this analysis, each dataset is evaluated under varying experimental conditions. To ensure consistency, the dataset and encoding scheme are fixed, while the evaluation focuses on the impact of different overlap sizes and drop-from strategies.

Titanic Full The DEA using TMH encoding achieves strong F1 scores on the Titanic dataset, with a peak of 0.76 for `DropFrom = Eve` and 0.68 for `DropFrom = Both` at overlap 0.9. This performance is largely driven by high precision (above 0.9) and improved recall for higher Recall, especially for Eve at 0.8 overlap (0.64). Compared to lower overlap scenarios, the model benefits dramatically from increasing training data. While Both initially underperforms Eve at 0.7–0.8, the gap narrows at higher overlap values.

No successful re-identifications are observed, neither fuzzy nor greedy matchers identify any records across overlaps. Despite solid structural learning, the model remains ineffective at perfectly linking back to individual identities on this dataset.

The hyper optimization validation scores slightly underestimate the training F1, but overall performance is close, particularly at higher overlaps.

In summary, TMH performs well on the Titanic dataset in terms of predictive accuracy, but fails to achieve individual re-identification under the evaluated conditions.

Fakename 1k The DEA evaluation on the TMH encoding for the fakename 1k dataset reveals several trends. When using the `DropFrom = Eve` configuration, the DEA consistently outperforms the baseline F1 score for overlaps higher than 0.4. Specifically, at overlaps of 0.6 and 0.8, the trained F1 score reaches approximately 0.54 and 0.58 respectively, clearly surpassing the baseline of around 0.2. This improvement is also reflected in the recall, which increases from about 0.11 at 0.4 overlap to over 0.42 at 0.8. Precision remains consistently high for Eve (above 0.95 across all overlaps), indicating that while not all matches are found, those that are predicted are likely correct. In contrast, the `DropFrom = Both` configuration shows weaker performance, only approaching the baseline F1 at the highest overlap (0.8), with notably lower recall and precision in all other settings. Interesting is the spike of precision at 0.8 overlap to nearly 0.9.

Despite the promising F1 scores, the actual re-identification rate remains at 0% across all overlaps and drop from strategies. None of the applied re-identification methods could recover any individual's true identity, indicating that while the model learns statistical patterns, it does not enable perfect linkage to plaintext identities.

When comparing trained F1 scores against the F1 score during hyper parameter optimization, all points lie below the $x = y$ line, suggesting that the trained model performs worse then during hyper paraemter optimization. This underestimation highlights that the hyper optimization metric may be conservative and not fully reflect the model's potential when applied to the entire dataset again.

Overall, these results show that the DEA can exploit structural patterns in TMH encodings effectively. However, the inability to re-identify individuals directly suggests that while aggregate information can be extracted, privacy at the individual level is maintained under these conditions.

Fakename 2k Compared to the 1k dataset, the DEA performance on the 2k version shows even clearer improvements when using `DropFrom = Eve`. At overlaps of 0.6 and 0.8, the trained F1 reaches 0.58 and 0.72 respectively, again clearly surpassing the baseline. Recall increases substantially (up to 0.61 at 0.8), and precision remains high (>0.9) except for a dip at 0.6. In contrast, the `DropFrom = Both` strategy performs poorly across all metrics, with only marginal gains in F1 and recall.

Similar to the 1k case, the re-identification rate remains 0% for all overlap levels and matching methods, reaffirming that the model exploits structural patterns but fails to link any record back to its plaintext identity. The generalization gap observed in the `hyper optimization vs. Full F1` plot persists, with trained performance consistently exceeding validation scores.

Overall, the trend confirms that increasing dataset size and overlap improves DEA effectiveness under the Eve setting, while direct re-identification remains infeasible.

Fakename 5k With the larger fakename 5k dataset, the DEA achieves its strongest results so far, especially for `DropFrom = Eve`. F1 peaks at 0.87 for overlap 0.6 and remains above 0.7 across higher overlaps. Precision nearly saturates at 1.0, while recall reaches 0.79. Compared to the 1k and 2k datasets, both the quality and stability of predictions improve. Even the `DropFrom = Both` configuration becomes effective, surpassing the baseline for overlaps ≥ 0.6.

For the first time, a small number of re-identifications occur: up to 1.2% (Eve) and 0.01% (Both) at overlap 0.6. These are still rare, but mark a turning point in the attack's ability to directly reconstruct individuals. The validation F1 (hyper optimization) continues to

underestimate final performance, though the gap is narrower than in smaller datasets.

These results indicate that with sufficient data volume and overlap, the DEA begins to break through into direct re-identification.

Fakename 10k With 10,000 records, the DEA reaches good performance. For `DropFrom = Eve`, the trained F1 climbs above 0.9 (overlap ≥ 0.6), and recall reaches 0.91, far exceeding the baseline. Precision saturates at nearly 1.0 across both drop strategies. Notably, `DropFrom = Both` also becomes effective with increasing overlap, achieving an F1 of 0.9 at overlap 0.8. Compared to the 5k dataset, all metrics improve further in both quality and consistency.

Importantly, re-identification becomes significant: up to 5.3% for Eve and 2.9% for Both at overlap 0.8, marking the first clear success in reconstructing individual identities. Greedy and fuzzy matching both contribute, with their combined result performing best. As in previous experiments, the validation F1 underestimates final performance, though the top points begin to approach the $x = y$ line.

Overall, this setting shows that the DEA not only recovers accurate group-level patterns but now also breaches individual privacy when data size and overlap are high.

Fakename 20k On the this dataset so far, the DEA reaches its highest effectiveness. Trained F1 exceeds 0.9 for both `DropFrom = Eve` and `DropFrom = Both` at overlap 0.8. Recall remains above 0.9 for Eve and 0.86 for Both, with near-perfect precision across all settings. Unlike smaller datasets, the Both strategy now reliably outperforms the baseline, even at lower overlap, demonstrating that DEA benefits from both increased data volume and redundancy.

Re-identification becomes alarmingly effective: Eve strategy enables recovery of up to 13.2% of individuals at overlap 0.8, while Both strategy yields up to 10.2%. The gap between matching strategies is consistent across all dataset sizes. The *greedy* matching always outperforms *fuzzy*, indicating that more confident matches are made based on graph-based reconstruction. The generalization gap between validation and full data F1 narrows for high-performing runs, but underfitting remains visible at lower overlaps.

In summary, at 20k records, the DEA no longer just learns structure, it reliably performs direct re-identification. The attack is most effective under the Eve strategy with high overlap, but Both is no longer a safe configuration as well.

Europerson For the euro person dataset, the DEA again achieves strong results across all overlaps and configurations. Trained F1 scores are consistently high, peaking at 0.95 for both `DropFrom = Eve` and `Both`, above the baseline. Recall and precision also remain high across all settings, suggesting stable and reliable model behavior. Unlike the fakename datasets, the Both strategy performs comparably well or even slightly better than Eve in terms of F1, indicating that the model generalizes robustly even when information is dropped symmetrically.

Re-identification is effective but concentrated at overlap 0.6, reaching 6.7% for Both and 6.1% for Eve. As in previous datasets, *greedy* matching consistently outperforms *fuzzy*. This reaffirms earlier observations that greedy matching is more effective for this task. The drop in re-identification rates at overlap 0.8, despite high model performance, suggests potential saturation effects or ambiguity in the match space.

However, the `hyper optimization vs. Full F1` plot shows nearly perfect correlation, suggesting excellent validation accuracy and minimal overfitting, marking a notable improvement over previous runs.

In summary, the DEA remains highly effective on the euro person dataset, with robust re-identification capabilities and strong generalization, especially under symmetric drop scenarios.

Fakename 50k

4.3.2. TSH

Titanic Full The TSH encoding yields consistently strong DEA performance on the Titanic dataset. F1 scores steadily increase with overlap, reaching 0.79 for `DropFrom = Eve` and 0.80 for `DropFrom = Both` at 0.9 overlap. This balanced performance across both configurations is supported by high precision values, above 0.95 for Both, and recall values climbing to over 0.74. Unlike TMH, which favored Eve at lower overlaps, TSH appears more stable and symmetric in terms of learned patterns.

Despite the good predictive performance, re-identification remains unsuccessful. No matches are found using fuzzy or greedy strategies, indicating that while structural properties are recoverable, they do not yet support direct re-identification under these conditions. The validation to full F1 correlation is especially tight at higher overlaps, with only a minor underestimation in one configuration.

In conclusion, TSH performs reliably on the Titanic dataset, yielding high precision reconstructions but no re-identifications, similar in privacy behavior to the other encoding methods tested.

Fakename 1k For the fakename 1k dataset, the DEA using TSH encoding achieves clear improvements over the baseline when overlap increases. Under `DropFrom = Eve`, the trained F1 reaches 0.72 at overlap 0.8, accompanied by a precision of 0.96 and recall of 0.60. The `DropFrom = Both` configuration lags behind at lower overlaps, but catches up by overlap 0.8 with an F1 of 0.65. This performance is again driven by strong precision values (0.97) and rising recall. Overall, TSH enables the model to learn discriminative patterns effectively, even with a relatively small dataset.

Despite these promising predictive results, no re-identifications were observed for any overlap or drop strategy. Both fuzzy and greedy matchers yielded a re-identifcation rate of 0%, indicating that while the model successfully reconstructs structure, individual identities remain protected in this setting.

The hyperparameter optimization validation scores follow the full data F1 trend with reasonable alignment, though slightly underestimated at high overlaps. These results suggest that even at small scale, TSH supports stable and accurate structure reconstruction, but without enabling perfect re-identification.

Fakename 2k With 2,000 records, the DEA performance improves significantly across both dropout strategies. For `DropFrom = Eve`, the trained F1 reaches 0.77 at overlap 0.6 and 0.82 at 0.8. The model achieves precision values above 0.9 while recall steadily increases, indicating that TSH enables the model to identify consistent structure as overlap grows. Interestingly, the `DropFrom = Both` configuration, while initially lagging, catches up at overlap 0.8, where it reaches an F1 of 0.84, slightly outperforming Eve in this setting.

Re-identification remains very limited. A few individual matches occur at higher overlaps, with a combined re-identification rate of 0.11% for Eve at overlap 0.6 and 0.29% for Both at

0.8. These are isolated and do not suggest a systematic breach. As seen previously, the successful re-identifications primarily stem from greedy matches, with fuzzy matching contributing marginally or not at all.

The hyperparameter optimization F1 estimates closely follow the full trained F1 values, particularly for the top-performing settings. Overall, the model demonstrates reliable learning behavior with growing structural accuracy, and only marginal re-identification leakage under favorable conditions.

Fakename 5k On the fakename 5k dataset, the DEA achieves robust structural reconstruction for both dropout strategies. For `DropFrom = Both`, the trained F1 increases steadily with overlap, peaking at 0.94 at overlap 0.8. Eve performs similarly, reaching an F1 of 0.89 at overlap 0.6, followed by a slight drop. Precision remains high across all settings (>0.95), while recall shows a consistent upward trend, reflecting stable learning behavior.

Re-identification becomes more prominent in this setting. For Eve, the combined re-identification rate reaches 2.1% at overlap 0.6, with contributions from both greedy and fuzzy matching. For Both, the re-identifcation rate peaks at 5.5% at overlap 0.8, the highest observed so far in this encoding scheme. Greedy matching again proves to be the dominant source of correct identifications, though fuzzy matching contributes increasingly at high overlap.

The hyperparameter optimization validation F1 aligns well with full-data F1, especially in the upper range, and reliably tracks the performance trend. Overall, TSH demonstrates its strongest re-identification capabilities in this setting so farh, especially under symmetric dropout and high overlap, confirming its vulnerability to structure-based leakage under DEA.

Fakename 10k On the fakename 10k dataset, the DEA again shows strong structure reconstruction under both dropout strategies. The trained F1 exceeds 0.9 for `DropFrom = Eve` at overlap 0.6 and reaches 0.95 for `DropFrom = Both` at overlap 0.8. Precision values remain consistently high (>0.95), while recall grows steadily, indicating stable learning behavior with increasing overlap. An exception occurs at overlap 0.6 under Both, where model performance drops sharply, likely due to instability in hyperparameter selection.

Re-identification becomes significant in this setting. For Eve, the combined re-identifciation rate peaks at 11.1% at overlap 0.6, with greedy matching again dominating. Both also yield a strong re-identifcation rate, reaching 5.6% at overlap 0.8. Fuzzy matching contributes modestly in all settings but never exceeds greedy performance.

Despite the outlier at 0.6 for Both, the hyperparameter optimization F1 generally tracks full-data performance reliably, especially for high-performing configurations. These results highlight that TSH poses a substantial re-identification risk when enough structural redundancy is present in the data, particularly under asymmetric dropout and moderate-to-high overlap.

Fakename 20k

Europerson On the euro person dataset, the DEA yields near-perfect reconstruction performance under both dropout strategies. Trained F1 scores exceed 0.98 at overlap 0.6 and 0.8, with both Eve and Both achieving precision and recall values above 0.97. This reflects that the TSH encoding retains highly discriminative structural patterns when applied to this dataset.

The strong performance is consistent and stable across overlap values, and the hyperparameter optimization validation scores closely track the final F1, indicating reliable model selection.

Re-identification rates are substantial in this setting. Under Eve, the combined re-ID rate reaches 23.1% at both 0.6 and 0.8 overlap. For Both, the rate follows closely with 21.9% at overlap 0.8. Greedy matching again contributes the majority of successful matches, but fuzzy matching begins to play a more significant role at higher overlaps. These rates indicate that TSH not only enables structural reconstruction but also allows for substantial identity recovery when the dataset structure is dense and consistent.

This result confirms that for structurally rich datasets, TSH can be highly vulnerable to dataset extension attacks. The re-identification potential under both symmetric and asymmetric dropout is pronounced and grows rapidly with overlap.

4.3.3. BF

Titanic Full The DEA achieves high trained F1 scores on the Titanic dataset, peaking at 0.83 for `DropFrom = Both` and 0.81 for `DropFrom = Eve` at overlap 0.8. Precision and recall both exceed 0.9 and 0.7 respectively in these top performing settings, indicating that the model successfully captures structural patterns from BF encodings. Compared to other datasets, performance is already strong at overlaps 0.7–0.9, and Both performs equally well or slightly better than Eve, suggesting that sufficient redundancy exists even under symmetric dropout.

Despite strong classification results, no successful re-identifications are observed across any setting. Both fuzzy and greedy matching produce 0% re-identifications, showing that although the model generalizes well, it does not achieve individual level deanonymization on this dataset.

Overall, while the model effectively learns link structures in the Titanic data, it falls short of re-identification, likely due to the dataset's size and feature variability.

Fakename 1k On the fakename 1k dataset, the DEA using BF encoding begins to show effective learning behavior under the `DropFrom = Eve` setting. The trained F1 score improves with overlap, reaching 0.68 at 0.8, far above the baseline.

This increase is driven by a steady rise in recall (from 0.04 to 0.53) and consistently high precision (up to 0.93). In contrast, the `DropFrom = Both` configuration fails to show meaningful improvements, with F1 scores plateauing below 0.2 and recall staying low despite overlap increases.

Despite strong F1 gains, no successful re-identifications are observed across any configuration or overlap. Both greedy and fuzzy matching yield 0% re-identification, indicating that while the model captures structural information, it cannot yet link encoded entries to plaintext identities.

The validation F1 (hyper optimization) underestimates full-data performance, especially for higher overlaps, consistent with trends seen in early-stage DEA setups. In summary, the BF DEA on the 1k dataset demonstrates its first signs of structural reconstruction under Eve only dropout, though without successful individual level re-identification.

Fakename 2k With 2,000 records, the DEA shows improvements over the 1k setup, especially for `DropFrom = Eve`. The trained F1 rises sharply with overlap, reaching 0.86 at 0.8. Recall increases to 0.8, and precision remains above 0.9, showing the model's growing capacity to identify correct patterns in BF encodings. In comparison, the `DropFrom = Both` strategy

improves slightly but remains less effective, with F1 peaking at 0.72 and lower recall across all overlaps.

For the first time, successful re-identifications occur: up to 1.5% for Eve and 0.3% for Both at overlap 0.8. As observed earlier, *greedy* matching begins to outperform *fuzzy* as overlap increases, indicating stronger exact-match results within the learned representations.

The hyper optimization metric still slightly underestimates final F1, but the gap narrows compared to the 1k dataset. Overall, the model transitions from pattern learning to early-stage identity reconstruction, particularly under asymmetric dropout and high overlap.

Fakename 5k On the 5,000-record dataset, the DEA shows strong and increasingly stable performance. Trained F1 peaks at 0.93 for `DropFrom = Eve` and 0.91 for `Both`, both at overlap 0.6, higher than in the 2k setting. Precision approaches 0.98–0.99, and recall exceeds 0.85, confirming the model's ability to learn accurate structures from BF encodings. Interestingly, at overlap 0.8, performance slightly drops for Eve, while Both remains stable, suggesting that excessive overlap might reduce contrast for learning.

Re-identification becomes meaningful: the Eve strategy yields up to 5.1% at overlap 0.6, and Both up to 1.6%. The dominant share of successful re-identifications again comes from *greedy* matching, confirming its higher effectiveness over *fuzzy*. These findings mark the transition from merely reconstructing structure to reliably mapping individual identities. As with earlier datasets, the hyper optimization validation score slightly underestimates final F1, though the correlation improves—especially for high-performing overlaps.

Overall, the 5k setting marks the first consistently successful phase of both structural learning and individual re-identification across dropout strategies.

Fakename 10k With 10,000 records, the DEA continues to scale effectively. Trained F1 scores reach 0.94 for `DropFrom = Eve` and 0.91 for `Both` at overlap 0.6, confirming stable structural learning across both asymmetric and symmetric dropout settings. Recall and precision remain very high (above 0.9) in all top-performing configurations. Performance slightly drops at overlap 0.8, indicating potential over-saturation in the match space.

Re-identification becomes substantial: Eve yields a combined re-identification rate of 11.1% and Both up to 7.9% at overlap 0.6. As in earlier datasets, *greedy* matching consistently outperforms *fuzzy*, with both contributing to the combined rate. This confirms the dominance of greedy strategies in reliably identifying true matches in BF space. Interestingly, re-identification performance dips again at 0.8 overlap, despite high F1, suggesting increased ambiguity among highly similar records. Validation F1 (hyper optimization) aligns closely with final results in high-performing runs, while low-overlap settings still show significant underestimation.

In summary, the 10k setting demonstrates mature DEA capabilities on BFs: both structure and identity are increasingly recovered, with Eve offering the highest success rates under intermediate overlap conditions.

4.3.4. Comparison between Encoding Schemes

4.4. Discussion

4.4.1. Methodological Considerations and Setup Validity

4.4.2. Interpretation of Results

4.4.3. Limitations and Practical Usefulness

4.4.4. Comparison with Other Approaches

5. Conclusion

5.1. Summary

This thesis investigated the vulnerabilities in PPRL systems with a particular focus on the feasibility and effectiveness of DEAs. As the integration of sensitive data across institutional boundaries becomes increasingly important, particularly in sectors such as healthcare, finance, and public security, the need for secure PPRL methods continues to grow. However, as this thesis has shown, widely used encoding techniques such as BF, TMH, and TSH remain vulnerable to GMAs and inference based re-identification methods like the DEA.

The primary research question guiding this work was whether it is possible to extend the capabilities of the GMA to re-identify not only overlapping individuals between plaintext and encoded datasets, but also individuals outside the intersection set. To this end, the DEA was proposed as a novel, learning-based extension of the GMA, capable of reconstructing plaintext information from encoded representations using ANNs trained on partially re-identified records. And from the results of the analysis it can be seen, that the DEA is indeed capable of reconstructing a significant portion of the original identifiers, even when the overlap between plaintext and encoded datasets is limited. Also the DEA outperforms baseline guessing methods most of the time, demonstrating the potential of learning-based inference attacks against PPRL systems.

The key contributions of this thesis are threefold. First, a modular and extensible DEA pipeline was developed, capable of adapting to different encoding schemes. Second, a tailored training and inference setup was implemented, leveraging PyTorch and GPU acceleration to efficiently train ANNs on n-gram prediction tasks. Third, a comprehensive experimental framework was established to benchmark the attack against multiple datasets, overlap levels, and encoding strategies, providing a foundation for comparative evaluation.

By building upon a refined GMA implementation, and extending it into a supervised inference-based attack, this work provides a concrete demonstration of how existing privacy preserving techniques may be overcome under realistic assumptions.

The DEA introduced in this thesis follows a modular and systematic six-step pipeline designed to generalize across encoding schemes. Beginning with a GMA to establish a baseline set of re-identified individuals, the DEA leverages these mappings as supervised training data for a neural model tasked with predicting n-gram structures in encoded representations.

To enable flexible experimentation, the attack pipeline was designed to accommodate three different encoding schemes: BF, TMH, and TSH. While the GMA itself is adaptable to the specific encoding, the DEA requires encoding-specific preprocessing routines, ANN architectures, and dataset representations. For instance, BF and TMH produce fixed-length bit vectors, which are converted directly into tensors, whereas TSH produces variable length integer sets that require normalization via one-hot encoding over a global dictionary of observed values.

A standardized data preparation process transforms the GMA output into structured datasets consisting of encoded inputs and corresponding multi-label n-gram labels. These datasets are then split into training, validation, and test subsets, with hyperparameter optimization per-

formed via Ray Tune and Optuna using the Dice coefficient as the optimization target. This metric was chosen for its ability to balance precision and recall in multi-label classification tasks, and for its interpretability in the context of reconstructing partial identifiers.

Each experimental run varies the attacker's knowledge by changing the dataset overlap between the plaintext and encoded databases, simulating realistic levels of auxiliary data availability. For every setting, a dedicated neural network is trained using the re-identified individuals as labeled data, and inference is performed on the remaining unmapped encodings. To assess model performance, both baseline frequency-based guessing and two deterministic reconstruction strategies, graph-based and dictionary-based, are used for comparative evaluation.

The core of the DEA's inference capability is a supervised ANN trained to predict the presence of character n-grams in encoded representations. This reframes the attack as a multi-label classification problem, where each output neuron corresponds to a possible n-gram, and the model predicts the probability of its inclusion in the plaintext. Despite the theoretical irreversibility of the underlying hash functions, the deterministic encoding structure allows the network to learn statistical associations across large training sets, thereby enabling probabilistic inference of plaintext components.

Nonetheless, the DEA cannot achieve perfect reconstruction due to the inherent limitations of similarity-preserving encoding schemes, most notably hash collisions and the lossy nature of BF. Instead, it operates under a probabilistic threat model, where the goal is to maximize partial reconstruction and re-identification success under practical constraints. The attacker is assumed to possess no cryptographic secrets or salts but may exploit knowledge of system design parameters and publicly available data, consistent with Kerckhoffs's principle and prior work on GMA based attacks.

To translate predicted n-grams into human interpretable identifiers, three reconstruction strategies were developed. The first method builds a directed graph from the predicted n-grams and attempts to find the longest coherent path through the character transitions. The second leverages auxiliary dictionaries of common first names, surnames, and birthdates to identify plausible matches via similarity scoring. The third strategy, based on large language models, uses a prompt based generative approach to reconstruct full identifiers from the predicted tokens. While powerful, the latter was excluded from the main evaluation due to its high variability, external dependencies, and limited reproducibility.

Each reconstruction strategy represents a different level of attacker sophistication, from deterministic heuristics to semantic matching. Together, they illustrate the extent to which encoded identifiers can be reversed into intelligible personal data using only structural and statistical clues, even when access to exact encoding parameters is restricted.

The findings and design of this thesis underscore a critical insight: even without access to cryptographic secrets or exact encoding parameters, an attacker can exploit structural patterns and statistical regularities within PPRL encoded data to perform effective re-identification. The DEA demonstrates how partial re-identifications obtained via a GMA can serve as a springboard for broader inference, ultimately undermining the core privacy guarantees of non-interactive, similarity-preserving linkage protocols.

This threat is particularly acute in real-world deployments where auxiliary data sources are abundant and overlap with target datasets is likely. As demonstrated in the DEA setup, even modest overlap rates can yield a meaningful training base for the neural model. Combined with efficient reconstruction strategies, this enables the attacker to recover a significant portion of the original dataset content, thereby transforming a partial breach into a systemic compromise.

The modularity of the DEA pipeline allows it to be extended to new encoding schemes, auxiliary data sources, or target attributes with minimal adjustments. Moreover, the use of accessible deep learning frameworks and off-the-shelf hardware illustrates that the technical barrier for executing a DEA is relatively low, further emphasizing the urgency for more robust PPRL mechanisms.

5.2. Future Work

While the DEA presented in this thesis establishes a practical and effective pipeline for reconstructing identifiers from encoded representations, several opportunities remain for future exploration, particularly in improving robustness under more challenging conditions. One of the central limitations of the current approach lies in its dependence on supervised learning, which requires labeled data produced by the GMA. In low-overlap scenarios, however, the number of successfully re-identified individuals becomes too small to support effective training, limiting the applicability of the DEA.

To address this, future work could explore unsupervised or semi-supervised learning strategies. For example, autoencoders or contrastive learning frameworks may be employed to learn latent representations of encoded records without relying on explicit labels. Alternatively, weak supervision through heuristics or synthetic data generation could help bootstrap model training in the absence of high GMA results. These approaches could increase the practical applicability of the DEA, particularly in real-world settings where overlap between auxiliary and target datasets can be low.

Another line of research could involve reversing the modeling direction altogether. Instead of predicting plaintext n-grams from encoded representations, one could design a model that learns to predict the corresponding encoded representation from a given set of n-grams. This would effectively invert the learning task, enabling a greedy reconstruction attack wherein an attacker iteratively constructs candidate plaintexts by adding one n-gram at a time. At each step, the model would generate a candidate encoding from the current n-gram set, and the attacker would retain the combination that maximizes similarity to the target encoded record. By fixing one n-gram and expanding the set greedily, the attacker could approximate the original encoding through iterative refinement, even in the absence of a direct mapping from encoding to n-grams.

In parallel, a reversed variant of the DEA could be explored, in which the attacker does not attempt to infer plaintext from encoded data directly, but rather aims to test specific hypotheses about potential individuals. In this setting, the attacker begins with a curated list of plausible names or identifiers—either generated using an LLM, derived from auxiliary datasets, or constructed via combinatorial enumeration. These candidates are then re-encoded using the trained model, and their encodings are compared against the set of unreidentified records using similarity metrics. This enables targeted queries such as: “Is a particular individual part of the dataset?” or “Which of these candidate names best matches the encoded record?”

This model-informed dictionary attack can operate in both brute-force and guided modes. In the brute-force variant, a large number of combinations are tested exhaustively. In the guided variant, candidate generation and ranking are refined using probabilistic heuristics or learned scoring functions. If the attacker possesses partial knowledge of the input distribution, such as region-specific name frequency statistics, known first names, or institutional contexts, this reversed strategy may be particularly effective. It is especially relevant for encoding schemes

that preserve structural locality or token frequency, such as BF or TMH, where approximate matches can yield meaningful meaning even in the presence of collisions or noise.

Finally, future work could explore a more principled connection between the encoding schemes and the architecture of the neural model itself. Since encoding processes such as BF or TMH can be described algebraically as matrix operations over hashed n-gram inputs, it may be possible to formalize the inverse mapping task as a linear or non-linear transformation. This formulation could inform the design of the ANN to allow for analytical derivation of network parameters, potentially reducing the reliance on extensive hyperparameter optimization. A deeper understanding of the encoding process as a differentiable or compositional function may also open the door to incorporating domain-specific inductive biases into the network, thereby improving learning efficiency and interpretability.

Beyond architectural enhancements, future work could also focus on algorithmic improvements to the DEA pipeline itself. One direction could be to establish a tighter integration between the GMA and the DEA, transforming the overall process into an iterative framework. In such a setup, the DEA could be applied not as a one-shot inference step, but as a looped refinement mechanism that feeds its most confident predictions back into the GMA module. These new pseudo-labels could then expand the training set for the next DEA iteration, gradually enlarging the set of re-identified individuals through a bootstrapping process. This would blend graph-based and learning-based re-identification strategies into a single adaptive attack.

Another important research direction is to systematically evaluate the robustness of the DEA in the presence of privacy-enhancing defenses. While this thesis focused on attacking standard encoding schemes without countermeasures, real-world systems could incorporate protective mechanisms such as n-gram dropout, salting, or the injection of synthetic noise to reduce re-identification risk. More advanced approaches, such as applying differential privacy during encoding, or using diffusion-based variants of BF, aim to further obfuscate the relationship between plaintext and encoded representations.

Complementary to algorithmic defenses, future work should also examine the generalization ability of the DEA across a broader range of datasets. Current evaluations rely on synthetically generated or cleaned datasets with relatively homogenous, Western-centric name distributions. Expanding this scope to include real-world datasets with naturally occurring noise, misspellings, and inconsistencies would better simulate deployment conditions and test model robustness. Additionally, integrating multilingual datasets or culturally diverse naming conventions would challenge the assumptions of the current pipeline and reconstruction strategies. For example, compound names, non-Latin scripts, or cultural variations in name structure may significantly impact n-gram frequency distributions and encoding patterns, potentially altering the effectiveness of both inference and reconstruction stages.

Another promising yet currently unexplored direction involves the integration of LLMs into the DEA attack pipeline. Although excluded from the core evaluation due to reproducibility concerns and computational cost, LLMs offer a powerful foundation for semantic reconstruction strategies. Future work could investigate cost-effective approaches to LLM integration, such as prompt tuning, knowledge distillation into smaller models, or constrained decoding methods tailored to the structure of PPRL encoded data. These techniques may offer a balance between the expressive power of generative models and the practical constraints of attack scalability.

Finally, future research should also consider the broader impact and ethical implications of DEA style attacks. While the primary aim of this thesis is to demonstrate the feasibility of inference-based attacks against PPRL systems, these methods could also be repurposed as auditing tools for evaluating the real-world resilience of deployed PPRL. In particular,

regulatory bodies, data custodians, or third-party evaluators could apply controlled versions of the DEA pipeline to assess whether deployed encodings are vulnerable to realistic adversarial inference under plausible auxiliary knowledge assumptions.

This naturally motivates a call for stronger privacy-preserving mechanisms and more standardized frameworks for risk evaluation in PPRL deployments. Current encoding schemes often lack formal guarantees against inference attacks, especially when the encoded data must remain linkable and non-interactive. Future research could explore ways to strengthen existing PPRL protocols through techniques such as probabilistic obfuscation, hybrid interactive schemes, or encoding structures designed to minimize semantic leakage. Ultimately, by highlighting concrete vulnerabilities and proposing paths toward mitigation, this line of work aims not only to critique existing systems, but also to contribute to the development of more robust and transparent PPRL technologies.

Bibliography

[AHS23] Frederik Armknecht, Youzhe Heng, and Rainer Schnell. "Strengthening privacy-preserving record linkage using diffusion." In: *Proceedings on Privacy Enhancing Technologies* (2023).

[BGF17] Facundo Bre, Juan Gimenez, and Víctor Fachinotti. "Prediction of wind pressure coefficients on building surfaces using Artificial Neural Networks." In: *Energy and Buildings* 158 (Nov. 2017). DOI: 10.1016/j.enbuild.2017.11.045.

[Blo70] Burton H Bloom. "Space/time trade-offs in hash coding with allowable errors." In: *Communications of the ACM* 13.7 (1970), pp. 422–426.

[Bro97] Andrei Z Broder. "On the resemblance and containment of documents." In: *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)*. IEEE. 1997, pp. 21–29.

[DKK+12] AD Dongare, RR Kharde, Amit D Kachare, et al. "Introduction to artificial neural network." In: *International Journal of Engineering and Innovative Technology (IJEIT)* 2.1 (2012), pp. 189–194.

[FS69] Ivan P Fellegi and Alan B Sunter. "A theory for record linkage." In: *Journal of the American Statistical Association* 64.328 (1969), pp. 1183–1210.

[Fou] The Linux Foundation. *PyTorch — pytorch.org*. https://pytorch.org. [Accessed 12-03-2025].

[GB10] Xavier Glorot and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks." In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2010, pp. 249–256.

[GSB+21] Fardin Ghorbani, Javad Shabanpour, Sina Beyraghi, Hossein Soleimani, Homayoon Oraizi, and M. Soleimani. "A deep learning approach for inverse design of the metasurface for dual-polarized waves." In: *Applied Physics A* 127 (Nov. 2021), p. 869. DOI: 10.1007/s00339-021-05030-6.

[HCR+16] Francisco Herrera, Francisco Charte, Antonio J Rivera, María J Del Jesus, Francisco Herrera, Francisco Charte, Antonio J Rivera, and María J del Jesus. *Multilabel classification*. Springer, 2016.

[HSW07] Thomas N Herzog, Fritz J Scheuren, and William E Winkler. *Data quality and record linkage techniques*. Vol. 1. Springer, 2007.

[IH18] Jim Isaak and Mina J Hanna. "User data privacy: Facebook, Cambridge Analytica, and privacy protection." In: *Computer* 51.8 (2018), pp. 56–59.

[KKM+14] Hye-Chung Kum, Ashok Krishnamurthy, Ashwin Machanavajjhala, Michael K Reiter, and Stanley Ahalt. "Privacy preserving interactive record linkage (PPIRL)." In: *Journal of the American Medical Informatics Association* 21.2 (2014), pp. 212–220.

[KM24] Jennifer King and Caroline Meinhardt. *Rethinking Privacy in the AI Era: Policy Provocations for a Data-Centric World*. White Paper, Stanford University Institute for Human-Centered Artificial Intelligence (HAI). 2024. URL: http://www.darkpatternstipline.org.

[MJ+01] Larry R Medsker, Lakhmi Jain, et al. "Recurrent neural networks." In: *Design and Applications* 5.64-67 (2001), p. 2.

[MK19] Karl Manheim and Lyric Kaplan. "Artificial intelligence: Risks to privacy and democracy." In: *Yale JL & Tech.* 21 (2019), p. 106.

[ON15] Keiron O'shea and Ryan Nash. "An introduction to convolutional neural networks." In: *arXiv preprint arXiv:1511.08458* (2015).

[PSZ+24] Aditi Pathak, Laina Serrer, Daniela Zapata, Raymond King, Lisa B Mirel, Thomas Sukalac, Arunkumar Srinivasan, Patrick Baier, Meera Bhalla, Corinne David-Ferdon, et al. "Privacy preserving record linkage for public health action: opportunities and challenges." In: *Journal of the American Medical Informatics Association* 31.11 (2024), pp. 2605–2612.

[RCS20] Thilina Ranbaduge, Peter Christen, and Rainer Schnell. "Secure and accurate two-step hash encoding for privacy-preserving record linkage." In: *Advances in Knowledge Discovery and Data Mining: 24th Pacific-Asia Conference, PAKDD 2020, Singapore, May 11–14, 2020, Proceedings, Part II 24*. Springer. 2020, pp. 139–151.

[RN16] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. pearson, 2016.

[SAH24] Jochen Schäfer, Frederik Armknecht, and Youzhe Heng. "R+R: Revisiting Graph Matching Attacks on Privacy-Preserving Record Linkage." In: *Proceedings of [Conference Name, if available]*. Available at: https://github.com/SchaeferJ/graphMatching. University of Mannheim. 2024.

[SBR09] Rainer Schnell, Tobias Bachteler, and Jörg Reiher. "Privacy-preserving record linkage using Bloom filters." In: *BMC medical informatics and decision making* 9 (2009), pp. 1–11.

[SMNP24] Vivek S Sharma, Shubham Mahajan, Anand Nayyar, and Amit Kant Pandit. *Deep Learning in Engineering, Energy and Finance: Principals and Applications*. CRC Press, 2024.

[SSA17] Sagar Sharma, Simone Sharma, and Anidhya Athaiya. "Activation functions in neural networks." In: *Towards Data Sci* 6.12 (2017), pp. 310–316.

[Smi16] Tanshanika T Smith. *Examining data privacy breaches in healthcare*. Walden University, 2016.

[Smi17] Duncan Smith. "Secure pseudonymisation for privacy-preserving probabilistic record linkage." In: *Journal of Information Security and Applications* 34 (2017), pp. 271–279.

[VCRS20] Anushka Vidanage, Peter Christen, Thilina Ranbaduge, and Rainer Schnell. "A graph matching attack on privacy-preserving record linkage." In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2020, pp. 1485–1494.

[VSCR17] Dinusha Vatsalan, Ziad Sehili, Peter Christen, and Erhard Rahm. "Privacy-preserving record linkage for big data: Current approaches and research challenges." In: *Handbook of big data technologies* (2017), pp. 851–895.

A. Auxiliary Information

Eidesstattliche Erklärung

Hiermit versichere ich, dass diese Abschlussarbeit von mir persönlich verfasst ist und dass ich keinerlei fremde Hilfe in Anspruch genommen habe. Ebenso versichere ich, dass diese Arbeit oder Teile daraus weder von mir selbst noch von anderen als Leistungsnachweise andernorts eingereicht wurden. Wörtliche oder sinngemäße Übernahmen aus anderen Schriften und Veröffentlichungen in gedruckter oder elektronischer Form sind gekennzeichnet. Sämtliche Sekundärliteratur und sonstige Quellen sind nachgewiesen und in der Bibliographie aufgeführt. Das Gleiche gilt für graphische Darstellungen und Bilder sowie für alle Internet-Quellen.
Ich bin ferner damit einverstanden, dass meine Arbeit zum Zwecke eines Plagiatsabgleichs in elektronischer Form anonymisiert versendet und gespeichert werden kann.

Datum	Marcel Mildenberger